ALMA MATER STUDIORUM - UNIVERSITÀ DI BOLOGNA
SEDE DI CESENA
FACOLTÀ DI SCIENZE MATEMATICHE, FISICHE E NATURALI
CORSO DI LAUREA MAGISTRALE IN SCIENZE E TECNOLOGIE INFORMATICHE

UN ALGORITMO PER LA SUPER RESOLUTION E LO ZOOMING DI UNA SINGOLA IMMAGINE CHE NE MANTIENE I CONTORNI E I DETTAGLI

Relazione finale in
Metodi Avanzati di Elaborazione di Immagini

Relatore
prof.ssa Laura Montefusco

Correlatore
dott.ssa Damiana Lazzaro

Presentata da
dott. Gioele Santi

Sessione II
Anno Accademico 2011/2012

Indice

Introduzione

Con l'evoluzione tecnologica degli ultimi anni, è diventato sempre più importante disporre di immagini ad alta risoluzione. La diffusione di formati di riproduzione sempre più dettagliati ha reso anche il grande pubblico consapevole di questa esigenza, ma non è solo l'industria dell'intrattenimento a beneficiare dell'alta risoluzione: in medicina immagini dettagliate possono migliorare l'efficienza delle tecniche diagnostiche ed avere una ricaduta positiva sulla qualità della vita; la cartografia basata su immagini digitali acquisite da satellite è diventata uno strumento sempre più importante e maggiori dettagli permettono di ricostruire mappe più accurate; l'astronomia ha nelle immagini una delle principali risorse di ricerca. Questi sono solo alcuni dei possibili esempi. Purtroppo per limiti tecnologici o per questioni di costi, non è sempre possibile utilizzare mezzi di acquisizione compatibili con le reali esigenze, perciò sono stati sviluppati numerosi algoritmi che permettono di aumentare la risoluzione delle immagini. Questi metodi, noti come metodi di super-resolution, vanno dai più semplici algoritmi di interpolazione implementati nei comuni software di grafica, sino a sofisticate tecniche basate sull'acquisizione di numerose immagini a bassa risoluzione. Ovviamente non sempre è possibile acquisire molteplici immagini, questo ha spinto a sviluppare nuovi metodi in grado di aumentare la risoluzione a partire da una singola immagine. Il problema in questo caso è la difficoltà nel ricostruire la grande quantità di informazioni mancanti dovuta ai limiti del processo di acquisizione. La recente teoria del Compressed Sensing permette di superare questo limite e ricostruire il segnale corretto sfruttando la caratteristica sparsità delle immagini.

Purtroppo la teoria del Compressed Sensing ha come presupposto la conoscenza della fase di acquisizione, che al contrario costituisce una delle principali incognite nell'ambito della super-resolution. In questo lavoro viene presentato una possibile soluzione al problema, che propone di integrare alla teoria del Compressed Sensing l'approccio a molti livelli di risoluzione, caratteristico della decomposizione wavelet, nel tentativo di individuare un modello

del processo di acquisizione sufficientemente flessibile da funzionare bene nel maggior numero di applicazioni.

Di seguito è riportato un breve sommario che illustra il percorso dei capitoli a seguire:

1. una breve trattazione sulla risoluzione delle immagini e sul processo di acquisizione digitale, con particolare attenzione rivolta ai disturbi ed alle sfocature che possono essere introdotte;
2. introduzione del modello matematico del problema della super-resolution e breve panoramica degli algoritmi comunemente utilizzati, da quelli basati su immagini multiple, sino a metodi di interpolazione adottati nel caso di immagini singole;
3. una trattazione che mostra come in determinati casi sia possibile superare i limiti della teoria dei segnali grazie al Compressed Sensing e come questo sia applicabile al campo delle immagini;
4. introduzione di un algoritmo per la super-resolution basato sulla teoria del Compressed Sensing (SR-CS), oltre alla parte teorica vengono illustrati alcuni dettagli implementativi ed in particolare la scelta del modello di acquisizione;
5. i risultati della sperimentazione condotta sui metodi implementati con una descrizione dei pregi e dei difetti individuati.

1 Risoluzione di un'immagine

1.1 Importanza dell'alta risoluzione

La risoluzione è una caratteristica intrinseca delle immagini, definita dal rapporto fra il numero dei punti utilizzati per la rappresentazione e le dimensioni della stessa. Le immagini ad alta risoluzione sono caratterizzate da un'alta densità di punti (o *pixel*) e quindi in grado di riprodurre una grande quantità di dettagli, anche più fini rispetto ad immagini meno dense di pari dimensioni. Con l'evoluzione tecnologica è diventato sempre più importante avere a disposizione immagini ad alta risoluzione e di buona qualità. L'esempio più immediato è sicuramente quello dell'industria dell'intrattenimento che negli ultimi anni ha abituato il pubblico a formati sempre più grandi e ricchi di dettagli. Ma le immagini ad alta risoluzione possono rivelarsi fondamentali anche in altri ambiti: in medicina un'immagine diagnostica con maggior dettaglio può essere determinante per avere una diagnosi corretta; le immagini satellitari più dettagliate possono essere utili per ottenere rilievi cartografici sempre più precisi.

Il metodo più intuitivo per ottenere immagini ad alta risoluzione è quello di utilizzare sensori caratterizzati da un gran numero di elementi sensibili (*fotodiodi*), aumentando così il *pixel-count* dell'immagine che, assieme ad altri fattori, ne determina la risoluzione. Questo può essere ottenuto in due maniere differenti: riducendo le dimensioni dei fotodiodi, aumentandone quindi la densità sul sensore fotografico; oppure mantenendo le dimensioni dei fotodiodi, ma aumentandone il numero e quindi anche la dimensione del sensore. Entrambe le strade sono state percorse, ma entrambe vanno incontro a limiti fisici invalicabili che ormai l'industria ha raggiunto: fotodiodi più piccoli catturano una quantità inferiore di luce, peggiorando il rapporto segnale rumore e quindi la qualità del risultato; sensori più grandi invece comportano una capacità elettrica superiore e quindi una ridotta velocità nel trasferimento delle cariche elettriche, dunque non sono efficaci. Inoltre i sensori fotografici non sono l'unico strumento per produrre un'immagine, ad esempio sui telescopi spaziali vengono a volte montati sensori molto simili, ma costruiti per essere sensibili ad una gamma di frequenze differente, non visibile

all'occhio umano. Mentre i sensori fotografici, data la grande domanda di mercato, hanno visto un abbattimento dei costi di produzione, questi altri sensori, molto più specifici, presentano tuttora prezzi molto elevati, che costringono spesso ad utilizzare dei chip di dimensioni e pixel-count ridotti rispetto le reali esigenze. Inoltre le immagini utilizzate nel campo della diagnostica sono prodotte da altre tecnologie (TAC, Risonanza Magnetica) e produrre immagini a risoluzioni più alte comporterebbe un aumento dei tempi di esposizione con evidenti disagi e rischi per i pazienti.

Questi fattori hanno reso necessario lo sviluppo di tecniche in grado di aumentare la risoluzione di un'immagine a posteriori. Queste sono note anche come tecniche di *super-resolution* e vanno dai metodi più semplici integrati nei programmi di elaborazione di immagini, sino a metodologie più sofisticate che permettono di ottenere immagini di alta qualità a partire da una o più immagini a bassa risoluzione, aprendo molteplici possibilità: dall'adattamento di immagini "piccole" a grandi formati di stampa o ai nuovi formati di riproduzione televisiva, sino allo zooming del dettaglio di un volto o di una targa da un filmato di sorveglianza.

1.2 Il processo di acquisizione delle immagini

Per meglio comprendere la natura della problematica, è meglio analizzare in che modo viene prodotta un'immagine digitale. Un dispositivo di acquisizione di immagini digitale è costituito da un apparato di lenti, un diaframma (che assieme alle lenti costituisce l'obiettivo), un otturatore (nel caso specifico delle fotocamere) ed un sensore. L'insieme delle lenti definisce il piano di messa a fuoco dell'immagine e convoglia la luce verso l'elemento sensibile all'interno della fotocamera, concentrando i fasci di luce provenienti da ogni singolo punto del soggetto inquadrato, su un singolo punto del piano focale. A seconda della tipologia di obiettivo considerato i fasci di luce vengono più o meno deviati, influenzando l'angolo di apertura dell'immagine catturata, ma rischiando di introdurre aberrazioni, che però in genere vengono compensate dal complesso di lenti interne allo stesso obiettivo. La qualità costruttiva dell'obiettivo, per quanto alta, non garantisce che la totalità della luce attraversi le lenti.

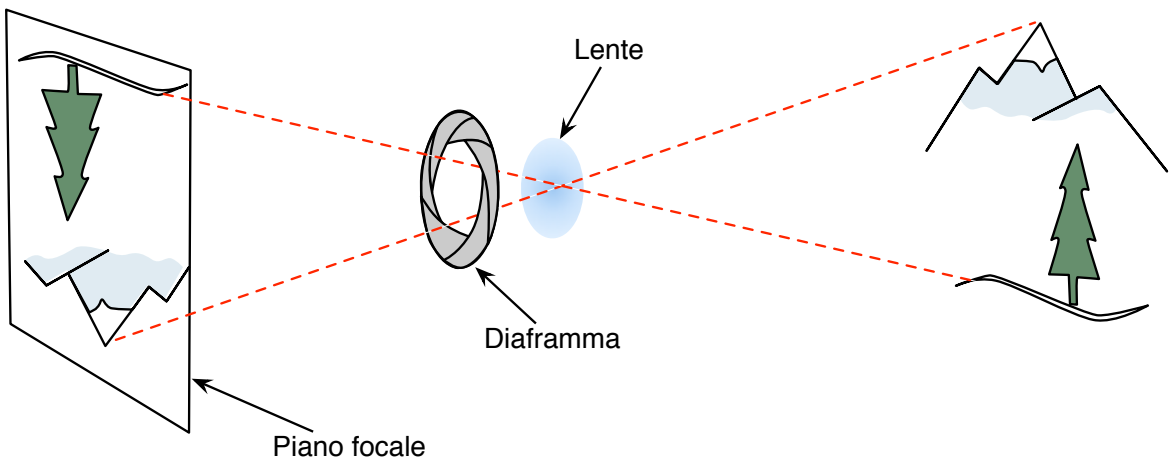


Fig1.1: illustrazione concettuale di un dispositivo per l'acquisizione di immagini.

Il diaframma si occupa di regolare la quantità di luce che raggiunge il sensore, ma poiché per farlo agisce sull'ampiezza dei fasci di luce, definisce anche la profondità di campo dell'immagine, ovvero il range centrato sul piano di messa a fuoco all'interno del quale i soggetti ripresi risultano ben definiti. Ad un diaframma più chiuso corrisponderà una quantità inferiore di luce ed una profondità di campo maggiore, ma un diaframma troppo chiuso causerà diffrazione con conseguente perdita di definizione. I soggetti che cadono al di fuori della profondità di campo proietteranno sul sensore un cono di luce più ampio del circolo di confusione del dispositivo, provocando quindi una sfocatura. Il termine utilizzato in letteratura è *blur*, che è pero una parola generica che indica differenti tipi di sfocatura, di natura e comportamento differente, quindi in fotografia questo particolare tipo di sfocatura viene indicato con il termine *bokeh* (dal giapponese *bokeru* - essere sfuocato).

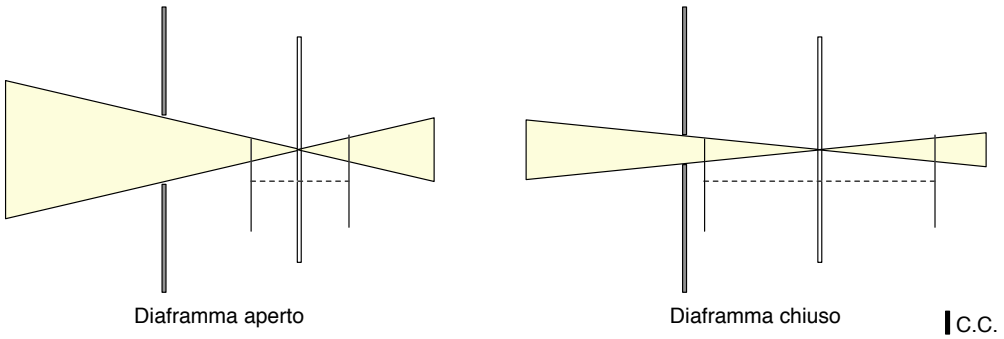


Fig 1.2: il circolo di confusione è il più piccolo cerchio distinguibile da una certa distanza. Se il fasci di luce proiettato è più piccolo del C.C. allora l'immagine risulta nitida, altrimenti il punto si espande in un disco. Chiudendo il diaframma aumenta la profondità di campo.

L'otturatore definisce il tempo di esposizione, scoprendo l'elemento sensibile per una certa quantità di tempo. In genere una scarsa quantità di luce (sia

ambientale, ma anche un diaframma particolarmente chiuso) rende necessari tempi di esposizione più lunghi, che però possono essere causa di un altro fenomeno di sfocatura, il *motion-blur*. Infatti se si prolunga il tempo di esposizione di un sensore, l'immagine finale sarà la sovrimpressione di una serie di immagini della stessa scena: eventuali soggetti in movimento presenteranno un alone lungo la direzione del movimento, mentre vibrazioni della fotocamera causeranno una sfocatura uniforme su tutta la scena.

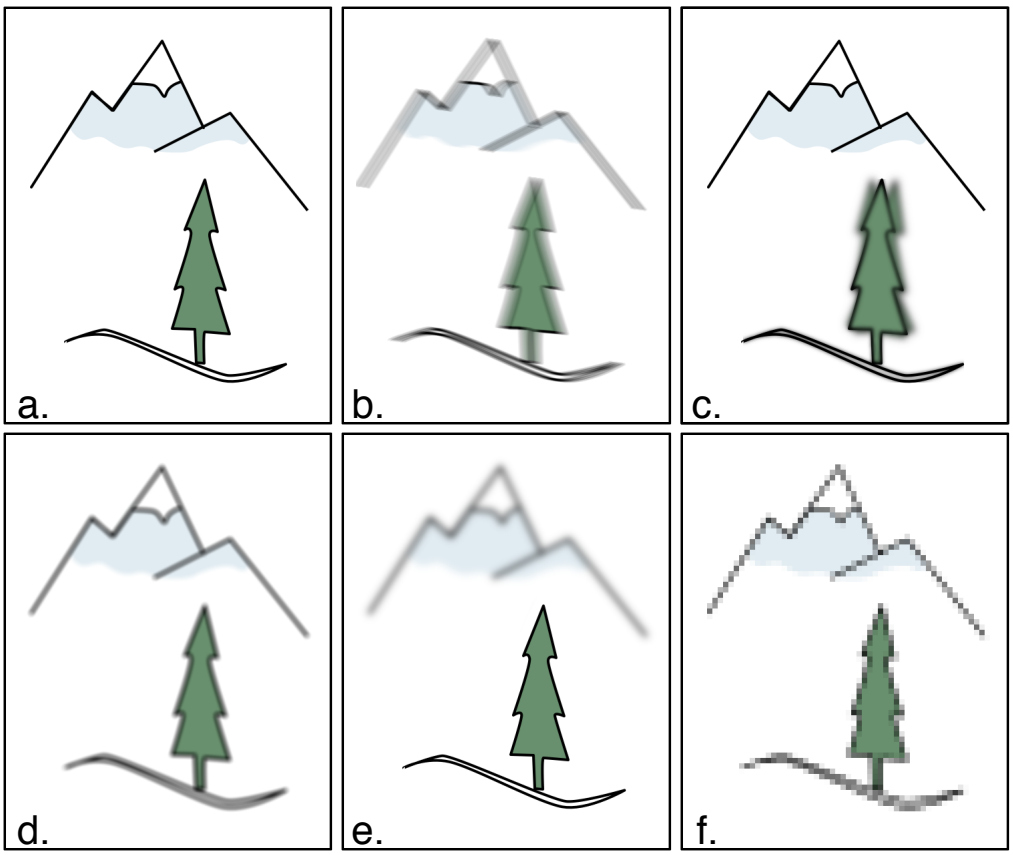


Fig 1.3: illustrazione di diverse tipologie di sfocatura: a) immagine completamente a fuoco; b) motion-blur dovuto al movimento della fotocamera; c) motion-blur dovuto al soggetto in movimento; d) mancanza di nitidezza dovuta alla diffrazione ; e) sfondo sfocato per la ridotta profondità di campo; f) difetti introdotti dalla PSF di un sensore a bassa densità.

Infine il sensore digitale, un reticolo di fotodiodi, ognuno dei quali quantizza un campione dell'immagine/segnale. In realtà i fotodiodi non sono in grado di distinguere il colore, ma solo la luminosità, quindi al sensore è anteposto un filtro, noto come *Color Filter Array* (CFA) che isola per ciascuno dei fotositi solo una singola componente cromatica. Il CFA più utilizzato è il pattern di Bayer, i cui filtri corrispondono allo standard RGB. Per ottenere l'immagine finale sarà necessario

applicare un processo di *demosaicing* che calcola le componenti cromatiche mancanti di ogni pixel in base ai valori circostanti.

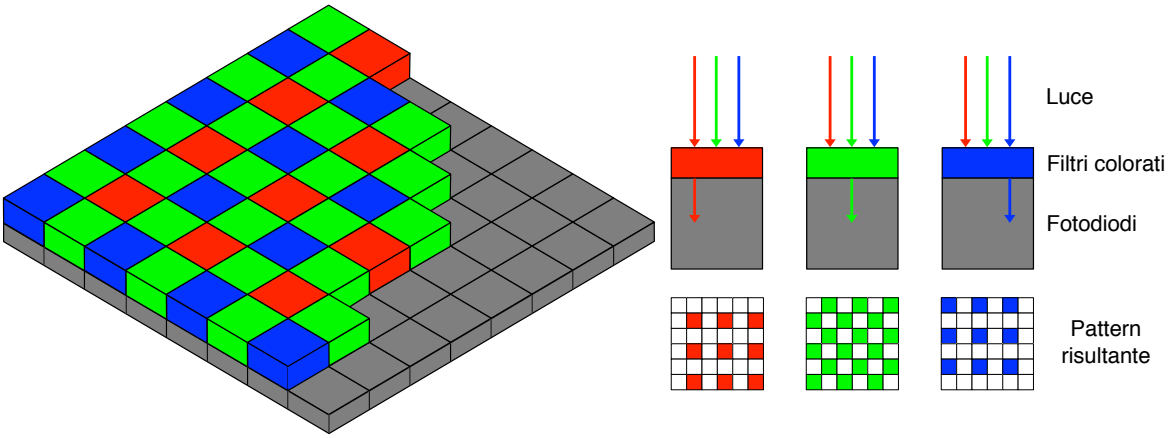


Fig1.4: esempio di pattern di Bayer. Ogni fotodiodo riceve solo una delle componenti cromatiche della luce. Esistono anche altre tipologie di CFA che considerano colori diversi dallo standard RGB.

La natura reticolare del sensore, lo rende soggetto all'effetto moiré, una particolare forma di *aliasing* che si verifica quando si inquadra un oggetto a sua volta caratterizzato da un reticolo di natura differente (per frequenza o inclinazione) da quella del sensore. Per arginare questo effetto le macchine fotografiche sono dotate di un filtro passa-basso (filtro anti-aliasing) che applica una sfocatura all'immagine, limitando di fatto l'ampiezza di banda del segnale entrante, creando i presupposti per l'applicazione del teorema di Nyquist-Shannon: la frequenza di campionamento di un segnale a banda limitata, deve essere almeno doppia rispetto all'ampiezza di banda.

La risoluzione dell'immagine risultante dipenderà dal numero di fotodiodi (pixel-count), dalla dimensione del sensore e dalla risolvenza dell'obiettivo. In particolare è importante l'abbinamento delle due componenti infatti la risoluzione finale sarà ottenuta con questa formula:

$$res = \frac{1}{\frac{1}{res_s} + \frac{1}{res_i}}$$

dove res è la risoluzione calcolata in linee per millimetro, res_s è la risoluzione del sensore, mentre res_i è la risoluzione del segnale in input passato attraverso l'apparato di lenti ed il filtro anti-aliasing. Quindi la risoluzione reale di un'immagine prodotta non sarà mai pari a quella promessa dalle dimensioni del sensore, ma

leggermente inferiore dato che, in alcuni casi, più linee di fotodiodi raccolgono l'informazione della stessa linea proiettata dall'obiettivo sul piano focale.

L'obiettivo ideale dei metodi di super-resolution è simulare l'acquisizione dell'immagine a partire da un sensore molto più denso, quindi aumentando il pixel-count (o pixel resolution). Idealmente ogni pixel di un'immagine raccoglie informazioni relative ad una porzione (per quanto piccola) della scena inquadrata. Ad ognuna di queste porzioni corrispondono infiniti punti nella realtà, ma ogni pixel ne riporta soltanto una media. Questa operazione corrisponde ad un'ulteriore forma di blur, legata alla PSF (*point spread function*) del sensore. L'individuazione di un modello sufficientemente preciso e flessibile, da descrivere tutti i vari fattori di sfocatura applicati all'immagine durante la fase di acquisizione, costituisce uno dei problemi centrali nel processo di incremento della risoluzione.

2 Aumentare la risoluzione

2.1 Modello matematico

Possiamo considerare l'immagine reale che si vuole registrare come una funzione continua a due dimensioni $F_C(x,y)$. Questa viene acquisita attraverso un reticolo discreto di fotodiodi, ottenendo una funzione discreta a due dimensioni $G(i,j)$. Ipotizziamo che esista un sensore più denso in grado di produrre direttamente un'immagine con risoluzione maggiore, il risultato sarà una nuova funzione 2D discreta $F(i,j)$. Per individuare la relazione che lega le due immagini, ne consideriamo la rappresentazione lessicografica: il vettore g di lunghezza m ed il vettore f di lunghezza n. È importante notare che $m < n$ in quanto l'immagine ad alta risoluzione è costituita da un numero maggiore di campioni. Il processo di acquisizione, già introdotto nel primo capitolo, invece può essere interpretato come l'applicazione di un filtro di blur ed un'operazione di decimazione. Sia quindi B la matrice nxn che contiene il filtro di blur. Esso riassume sia la PSF relativa al sensore che tutti gli altri fattori di sfocatura che potrebbero essere stati applicati in fase di acquisizione. Sia D mxn la matrice di decimazione che, applicata ad un'immagine, ne riduce effettivamente il numero dei pixel. Allora il processo di acquisizione potrà essere descritto matematicamente dalla seguente equazione:

$$g = D \cdot B \cdot f \tag{2.1}$$

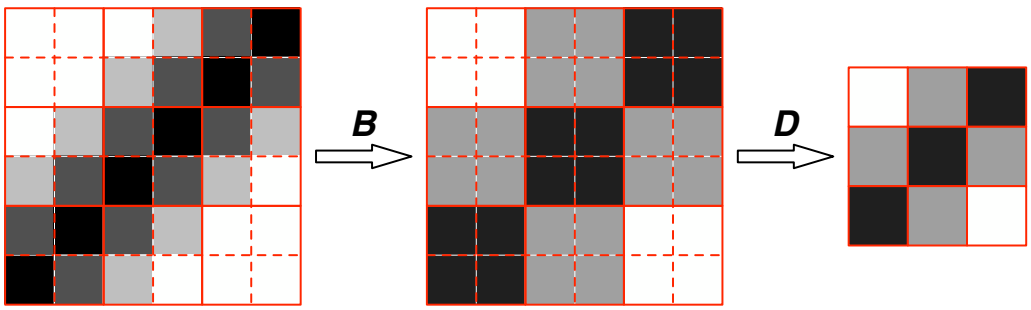


Fig 2.1: rappresentazione intuitiva del processo di acquisizione. All'immagine viene applicato un filtro passa basso B che rimuove le alte frequenze (in questo caso una media dei 4 pixel circostanti) mentre l'operatore di decimazione seleziona solo una porzione dei pixel originali riducendo il numero dei punti.

Sia $H = D{\cdot}B$, allora il modello di acquisizione può essere riscritto come:

$$g = H \cdot f \tag{2.2}$$

in cui H è una matrice mxn con $m < n$. Questa formulazione contiene implicitamente anche la definizione del problema dell'aumento di risoluzione, che consiste nel risolvere il sistema lineare

$$H \cdot f = g \tag{2.3}$$

Purtroppo questo è un problema indeterminato, poiché il numero di equazioni è inferiore al numero delle incognite ed esistono infinite soluzioni ammissibili. A questo si aggiunge il fatto che l'immagine acquisita g può essere affetta da rumore, influenzando negativamente la stabilità del problema, ed il fatto che nella maggior parte dei casi la matrice B non è nota. Quindi risolvere il problema della super-resolution implica anche la soluzione di un problema di *restoration* e di *blind-deconvolution*.

2.2 Utilizzare più immagini

La maggior parte dei metodi sviluppati compensa le scarse informazioni del modello illustrato aumentando il numero di immagini a bassa risoluzione, ad esempio considerando una serie di acquisizioni successive dello stesso dispositivo (anche un filmato) o più immagini riprese da punti di vista differenti. Se le acquisizioni rispettano alcuni requisiti, allora si riesce ad integrare il sistema di equazioni ricavando un problema determinato, quindi risolvibile. In particolare il modello può essere riformulato nel seguente modo:

$$g_k = D \cdot B_k \cdot W_k \cdot f \qquad 1 \leq k \leq p \tag{2.4}$$

aggiungendo la matrice di distorsione W_k che descrive le variazioni delle immagini rispetto ad una di riferimento (traslazioni e rotazioni globali e locali). Ricostruendo le distorsioni applicate è possibile ricondurre il problema ad un sistema sovradeterminato, risolvibile con il metodo dei minimi quadrati. Purtroppo però in alcuni casi il sistema così ottenuto resta comunque mal condizionato, rendendo necessaria l'adozione di tecniche di regolarizzazione. In genere le informazioni disponibili vengono integrate con delle conoscenze a priori sull'immagine ad alta

risoluzione, quindi si cerca una soluzione che minimizzi una determinata funzione obiettivo. Un esempio è il metodo CLS (*Constrained Least Squares*) che ricerca fra tutte le possibili soluzioni quella con maggiore *smoothness* [6].

Un'altra tecnica sempre basata sull'acquisizione di più immagini è quella dell'interpolazione non uniforme. Per prima cosa vengono stimate le distorsioni applicate in fase di acquisizione, stabilendo la posizione dei punti delle immagini a bassa risoluzione rispetto all'ipotetico reticolo ad alta densità che si vuole ottenere. Infine i pixel dell'immagine ad alta risoluzione sono ottenuti come media pesata dei punti più vicini. È essenziale che le variazioni fra un'immagine a bassa risoluzione e l'altra siano a livello sub-pixel, per poter catturare quelle informazioni che non possono essere fisicamente registrate dal sensore. Questa soluzione è anche implementata fisicamente su alcuni modelli professionali di macchine fotografiche, progettati esclusivamente per la ripresa di scene statiche: anziché scattare una singola foto vengono scattate una sequenza di foto traslando il sensore in varie direzioni.

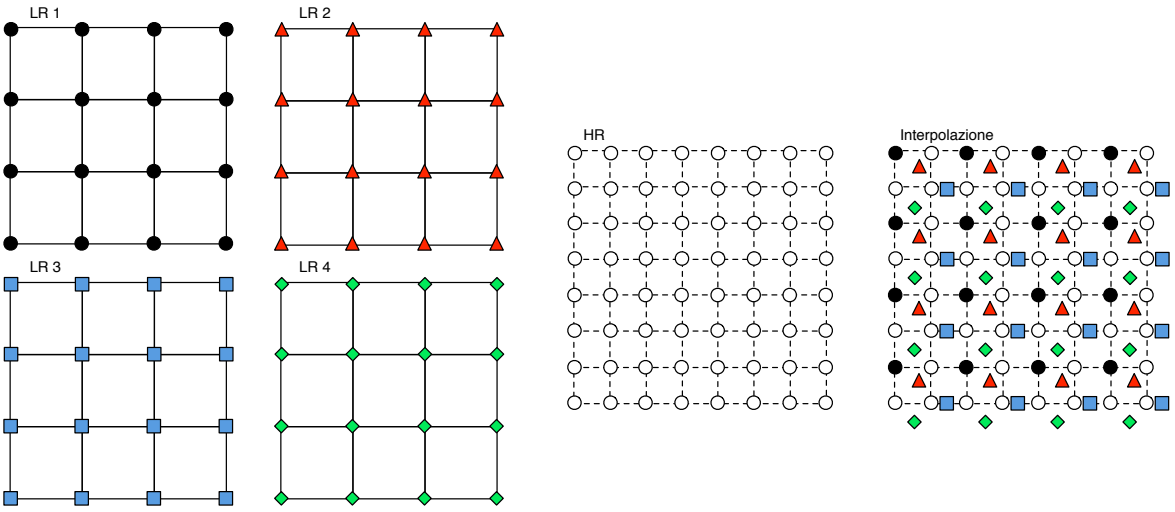


Fig.2.2: esempio di interpolazione non uniforme. Sono acquisite quattro immagini a bassa risoluzione; viene stimato lo spostamento fra le quattro immagini ed i pixel sul reticolo più denso vengono ottenuti per interpolazione.

Un metodo differente consiste invece nello sfruttare la correlazione fra la trasformata discreta di Fourier delle immagini a bassa risoluzione e la trasformata continua dell'immagine reale. Considerando in particolare la proprietà di shifting della trasformata di Fourier ed il presupposto che l'immagine ad alta risoluzione sia a banda limitata, si può riscrivere il modello di acquisizione nel modo seguente:

$$\hat{g}_k = \Phi_k \cdot \hat{f} \tag{2.5}$$

Dove il segno ˆ contraddistingue i vettori contenenti i coefficienti delle trasformate di Fourier delle rispettive immagini e la matrice Φ campiona solo alcuni dei coefficienti. Questo approccio ha il vantaggio di essere parallelizzabile, ma anche il limite di funzionare solo se nell'acquisizione sono applicate traslazioni globali e blur uniforme.

2.3 Immagine singola

I metodi che utilizzano immagini multiple riescono spesso a dare risultati eccellenti, ma non sempre è possibile effettuare più di un'acquisizione, dunque in questi casi l'aumento di risoluzione è inevitabilmente un problema mal posto. Esistono comunque diverse strategie possibili per aumentare la risoluzione delle immagini, le due principali famiglie sono:

- algoritmi di interpolazione;
- algoritmi basati sull'apprendimento.

Fra questi l'interpolazione è sicuramente l'approccio più intuitivo, anche se non è universalmente considerato un metodo di super-resolution in quanto non in grado di recuperare la alte frequenze dell'immagine [6]. Infatti l'obiettivo principale dei metodi di interpolazione è produrre un'immagine gradevole preservando le caratteristiche geometriche dell'immagine originale. Quindi sono l'ideale per l'adattamento di immagini di piccole dimensioni a grandi formati di riproduzione, ma costituiscono anche un punto di riferimento comune per la valutazione di altri algoritmi per la super-resolution.

Gli algoritmi di interpolazione non coinvolgono il modello matematico visto in precedenza, infatti l'idea alla base di questi metodi prevede semplicemente di rimappare i pixel dell'immagine a bassa risoluzione all'interno di una matrice di dimensioni maggiori e di calcolare i punti mancanti in base a quelli noti. I metodi più semplici sono quelli lineari, che applicano su tutta l'immagine lo stesso tipo di interpolazione, senza considerarne le caratteristiche principali come gli *edge*. Questo produce una serie di artefatti all'interno dell'immagine (scalettatura, blurring, aloni) che si concentrano attorno a bordi (alte frequenze). Di questi metodi fanno parte: l'interpolazione *nearest-neighbour*, che si limita a replicare il valore dei pixel più vicini; l'interpolazione bilineare, che ricava i punti mancanti

attraverso una media dei quattro pixel noti più vicini; l'interpolazione bicubica, che calcola i nuovi pixel come media dei sedici pixel noti più vicini.

Fig 2.3: Quattro metodi di interpolazione lineare confrontati con l'immagine originale (OR). nearest neighbour (NN) introduce la caratteristica scalettatura, mentre l'interpolazione bilineare (BL) e bicubica(BC) introducono sfocatura nell'immagine (fonte immagine [8])

Altri metodi di interpolazione sfruttano le informazioni relative ai bordi per migliorare il risultato finale, questi sono noti come metodi *edge-directed*. Uno dei più citati in letteratura è sicuramente NEDI (*New Edge-Directed Interpolation*)[7], che stima l'orientazione degli edge con il calcolo della covarianza e quindi adotta due strategie di interpolazione differenti a seconda che il punto da determinare si trovi o meno lungo un bordo: nelle aree uniformi viene applicata una semplice interpolazione bilineare, mentre lungo gli edge il pixel è ricavato con la seguente formula:

$$Y_{2i,2j} = \sum_{k=0}^{1}\sum_{l=0}^{1} \alpha_{2k+l} \cdot Y_{2(i+k),2(j+l)} \tag{2.6}$$

Per determinare i pesi associati a ciascun pixel noto, viene definito il vettore

$$\overrightarrow{y} = \begin{bmatrix} y_{2i-2,2j-2} & y_{2i-2,2j} & \dots & y_{2i+4,2j+4} \end{bmatrix}^T \tag{2.7}$$

contente i valori dei punti nell'intorno del pixel da interpolare definito da una matrice quadrata M. Sia C la matrice avente per ogni riga i quattro vicini diagonali di ciascun elemento nel vettore y:

$$C = \begin{bmatrix} y_{2i-4,2j-4} & y_{2i-4,2j} & y_{2i,2j-4} & y_{2i,2j} \\ y_{2i-4,2j-2} & y_{2i-4,2j+2} & y_{2i,2j-2} & y_{2i,2j+2} \\ \dots & \dots & \dots & \dots \\ y_{2i+2,2j+2} & y_{2i+2,2j+6} & y_{2i+6,2j+2} & y_{2i+6,2j+6} \end{bmatrix} \tag{2.8}$$

L'immagine è modellabile come un processo gaussiano localmente stazionario, ossia esistono regioni di immagine sufficientemente piccole da assumere che, se presente, ci sia un unico edge uniforme. Quindi i pesi α vengono calcolati in modo da minimizzare l'errore quadratico medio (MSE) per ciascuno dei punti in y:

$$MSE = \left\| \overrightarrow{y} - \overrightarrow{\alpha} C \right\|^2 \quad si\ pone \quad \frac{\delta(MSE)}{\delta \overrightarrow{\alpha}} = 0 \tag{2.9}$$

Il vettore è quindi ricavato con la formula:

$$\overrightarrow{\alpha} = (C^T C)^{-1} (C^T \overrightarrow{y}) \tag{2.10}$$

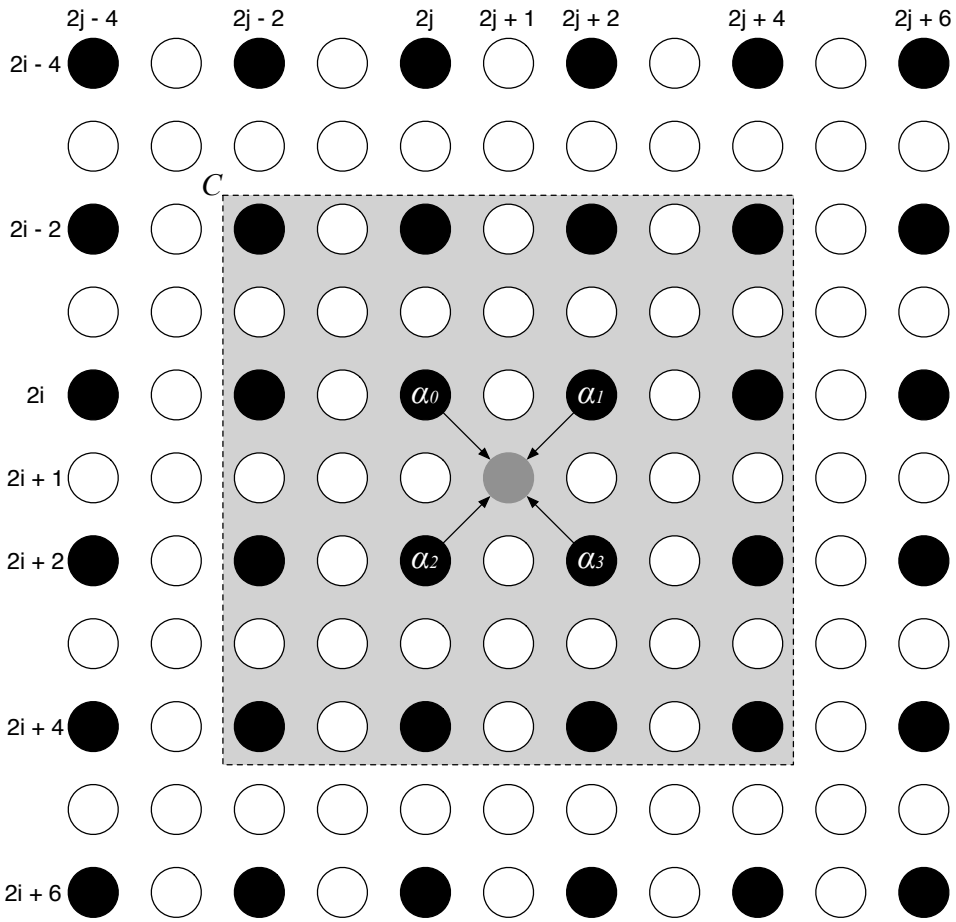


Fig. 2.4: Funzionamento di NEDI: il pixel in posizione (2i+1, 2j+1) è calcolato con la media pesata dei quattro punti circostanti (contrassegnati con i rispettivi pesi α). Inoltre è evidenziata la matrice C presentata nell'equazione (2.8) Dopo un primo passo di interpolazione, viene applicata la stessa procedura una seconda volta ruotando di 45° per calcolare i pixel non determinati al primo passaggio.

Comunque NEDI presenta alcune criticità, infatti la matrice quadrata M tende ad enfatizzare gli edge verticali, orizzontali ed inclinati di 45°, inoltre sua dimensione è fissata a priori e non si adatta alle esigenze di ogni particolare area:

una maschera piccola può ridurre la sfocatura a scapito della robustezza. INEDI (*Improved NEDI*) [8] è un algoritmo successivo che cerca di risolvere questi inconvenienti adottando una maschera M circolare, la cui dimensione viene aumentata iterativamente fino ad ottenere un risultato ottimale nel calcolo del MSE. Altri accorgimenti adottati sono:

- alcuni metodi di selezione dei punti interni alla maschera M, in modo da rendere il metodo più robusto rispetto al rumore;
- il calcolo del vettore α con la seguente formula:

$$\overrightarrow{\alpha} = C^{+}(C^{T}\overrightarrow{y}) \tag{2.11}$$

dove $C+$ indica la pseudo-inversa di Moore-Penrose, che sostituisce il calcolo della matrice C^TC, spesso mal condizionata.

Altri metodi di interpolazione edge-directed invece stimano l'orientazione dei bordi con il calcolo della derivata seconda. In particolare ICBI (*Iterative Curvature Based Interpolation*) [9] applica un primo passo di interpolazione adottando l'algoritmo FCBI (*Fast Curvature Based Interpolation*), che ricava i pixel come media di soli due punti noti secondo questa regola:

$$y_{2i+1,2j+1} = \begin{cases} (y_{2i,2j+2} + y_{2i+2,2j})/2 & se\ \tilde{y}^{(11)}_{2i+1,2j+1} > \tilde{y}^{(22)}_{2i+1,2j+1} \\ (y_{2i,2j} + y_{2i+2,2j+2})/2 & altrimenti \end{cases} \tag{2.12}$$

dove

$$\begin{aligned} \tilde{y}^{(11)}_{2i+1,2j+1} = {} & y_{2i,2j-2} + y_{2i+2,2j} + y_{2i+4,2j+2} - 3y_{2i,2j} - \\ & -3y_{2i+2,2j+2} + y_{2i-2,2j} + y_{2i,2j+2} + y_{2i+2,2j+4} \end{aligned} \tag{2.13}$$

$$\begin{aligned} \tilde{y}^{(22)}_{2i+1,2j+1} = {} & y_{2i-2,2j+2} + y_{2i,2j} + y_{2i+2,2j-2} - 3y_{2i,2j+2} - \\ & -3y_{2i+2,2j} + y_{2i,2j+4} + y_{2i+2,2j+2} + y_{2i+4,2j} \end{aligned} \tag{2.14}$$

sono le stime della derivata seconda nelle due direzioni diagonali. Infine adotta una procedura iterativa che applica dei fattori di correzione ad ogni pixel in modo da ottenere edge regolari senza introdurre eccessivo smoothing o artefatti nell'immagine.

Gli algoritmi basati sull'apprendimento sfruttano un database di immagini a vari livelli di risoluzione per ricostruire i dettagli assenti nelle immagini a bassa risoluzione. Questo approccio si basa sull'osservazione che, nel meccanismo di visione degli animali, la mancanza di informazioni viene compensata con altri elementi già noti per esperienza. Il principale difetto è sicuramente la non assoluta veridicità dei dettagli ricostruiti (comunque non noti). Inoltre la costruzione di un database sufficientemente completo per rendere il metodo efficace su una qualunque immagine è un processo complicato. Un metodo recente ha cercato di ovviare la necessità di un database, ricavando le informazioni necessarie alla ricostruzione dalla stessa immagine a bassa risoluzione. Infatti un'osservazione statistica dimostra come delle patch sufficientemente piccole si ripetano a diversi livelli di risoluzione all'interno delle immagini naturali [10].

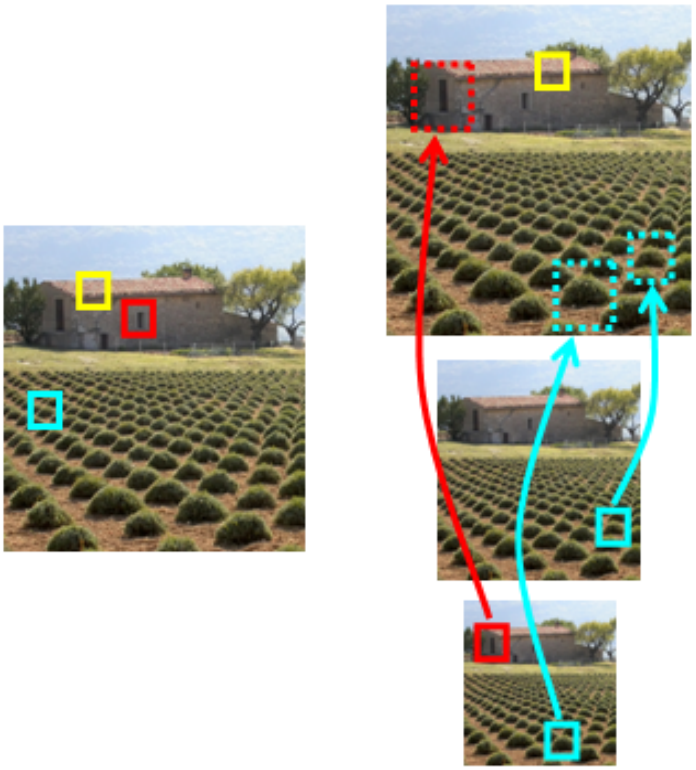


Fig 2.5: un esempio della ricorrenza delle patch a diversi livelli di risoluzione, principio sfruttato per la super-resolution basata sull'apprendimento (fonte immagine [10]).

3 Compressed Sensing

3.1 Superare il teorema di Nyquist Shannon

Come abbiamo potuto vedere, fra le tecniche per la super-resolution ad immagine singola ben poche tengono in considerazione la definizione matematica del problema, preferendo altre strategie più intuitive che però non hanno come obiettivo quello di ricostruire nel modo più fedele possibile l'immagine originale ad alta risoluzione, ma semplicemente rendere gradevole un'operazione di *upscaling*. Se riconsideriamo la formulazione matematica del problema, è evidente che la difficoltà più grande è l'esistenza di infinite soluzioni ammissibili per il sistema di equazioni (2.3). Dal lato pratico questo significa che il segnale a bassa risoluzione può essere stato prodotto da un'infinità di possibili immagini, delle quali solo una è quella corretta. Questo problema è noto come aliasing: una frequenza di campionamento troppo bassa, che nel caso delle immagini corrisponde appunto ad una bassa risoluzione, non permette di risalire al segnale originale, che è indistinguibile dagli altri possibili segnali.

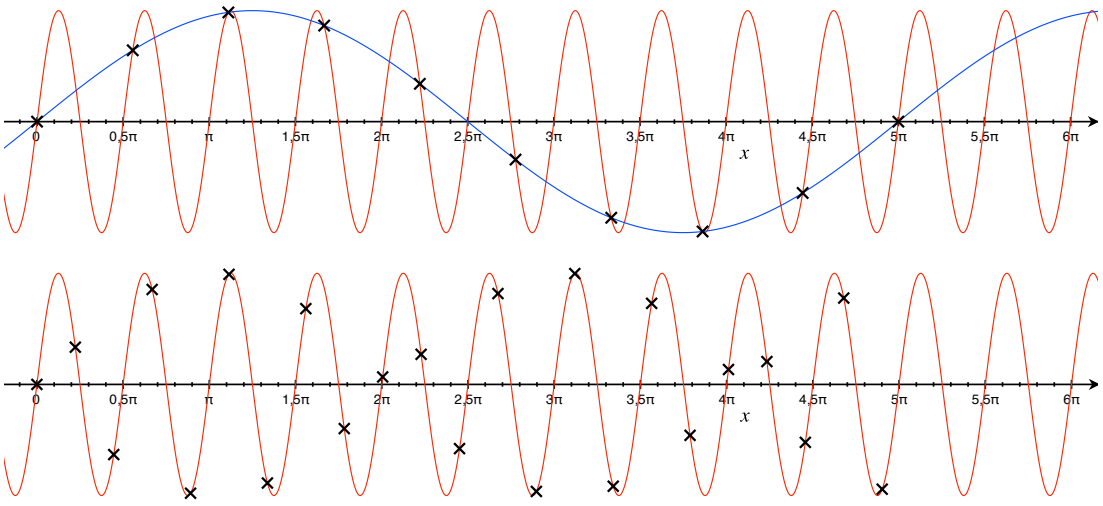


Fig 3.1: un'illustrazione del principio dell'aliasing, nel primo caso i campioni non sono sufficienti per comprendere se a generarli sia stato il segnale blu o quello rosso, nel secondo caso viene rispettato il teorema di Nyquist-Shannon ed è possibile determinare il segnale senza ambiguità.

Come abbiamo già visto nel primo capitolo, la teoria del campionamento, ed in particolare il teorema di Nyquist-Shannon, dice che un segnale a banda limitata, la

cui frequenza massima è f_M, per essere riprodotto senza aliasing deve essere campionato con una frequenza maggiore o uguale a $2f_M$. Sempre nel capitolo 1 abbiamo visto come il presupposto del segnale a banda limitata, sia dato dalla presenza di un filtro anti-aliasing che, applicando una sfocatura, taglia le frequenze più alte. Quindi possiamo interpretare l'immagine a bassa risoluzione di partenza come il risultato di un campionamento con frequenza più bassa rispetto a quella richiesta dal teorema di Nyquist-Shannon.

A questo punto la teoria classica dell'elaborazione dei segnali escluderebbe ogni possibilità di individuare il segnale originale, ma una recente teoria nota come *Compressed Sensing* (o *Compressive Sensing*, Campionamento Compressivo) permette di superare questo ostacolo nel caso in cui il segnale da ricostruire sia sparso. Le immagini naturali non sono esattamente segnali sparsi, ma hanno la caratteristica di essere compressibili: dal punto di vista matematico, di tutte le informazioni raccolte nel processo di acquisizione, molte risultano superflue per la riproduzione e quindi eliminabili senza alterare in maniera percettibile l'immagine. Gli algoritmi di compressione funzionano tutti su questo principio: l'immagine è convertita in un'altra base di rappresentazione in cui, di tutti i coefficienti utilizzati per rappresentarla, vengono mantenuti solo quelli superiori ad una certa soglia ed eliminati i restanti, senza per questo perdere informazioni significative dell'immagine. Dunque possiamo assimilare le immagini a segnali sparsi. La teoria del Compressed Sensing semplicemente cerca di unire in un'unica fase i passi di acquisizione del segnale e compressione, in modo da ridurre a monte il numero di campioni necessari per ottenere un segnale ricostruito di ottima qualità.

3.2 Teoria del Compressed Sensing

Si consideri un generico segnale x di dimensione n:

$$x = \begin{pmatrix} x_1 \\ ... \\ ... \\ ... \\ x_n \end{pmatrix} \quad t.c. \quad x_i \in \Re \quad \forall i = 1, ..., n \tag{3.1}$$

Il requisito fondamentale per ridurre il campionamento è la sparsità del segnale, ossia che degli n coefficienti che lo descrivono solo $k < n$ siano non nulli. Se il

segnale x non è sparso sarà necessario individuare una base di rappresentazione adeguata Ψ tale che:

$$x = \Psi \cdot s \quad ovvero \quad x = \sum_{i=1}^{n} s_i \cdot \psi_i \tag{3.2}$$

con ψ_i linearmente indipendenti. Il vettore s è la rappresentazione del segnale nella nuova base e si dice k-sparso se vale:

$$\|s\|_0 \leq k \tag{3.3}$$

dove la norma l_0 conta gli elementi non nulli di s. Quindi il vettore x può essere ottenuto attraverso la combinazione lineare di solo k degli n vettori base:

$$x = \sum_{\forall i \ t.c. \ s_i \neq 0} s_i \cdot \psi_i \tag{3.4}$$

La teoria può essere però applicata anche a segnali compressibili, infatti se solo k coefficienti sono significativi, gli n-k coefficienti più piccoli possono essere eliminati approssimando il segnale originale con un segnale k-sparso. Per questa ragione, nel caso particolare delle immagini, possono essere adottate le basi in genere utilizzate per la compressione, che hanno la caratteristica di essere sparsificanti. Più in generale, come poi vedremo, si considereranno domini in cui le immagini presentano maggiore sparsità, come ad esempio quello del gradiente.

L'acquisizione è una fase molto importante per la teoria del Compressed Sensing, se infatti è possibile ridurre il numero di misurazioni necessarie per la ricostruzione corretta del segnale, è però necessario effettuare questi campionamenti rispettando alcune condizioni. Innanzitutto il dominio in cui avviene l'acquisizione deve essere diverso da quello in cui il segnale è sparso(incoerenza del campionamento). Inoltre poiché il segnale registrato sarà un vettore y di dimensioni $m < n$, le informazioni raccolte non potranno essere un sottoinsieme dei coefficienti x_i, infatti in tal caso sarebbero davvero insufficienti per la ricostruzione. La chiave della teoria è acquisire una serie di funzioni lineari del segnale x, registrando quindi più informazioni. Queste funzioni sono definite da m

vettori $\phi_j \in \Re^n$ e la fase di campionamento viene realizzata calcolandone il prodotto scalare per il segnale x:

$$y_j = \langle x, \phi_j \rangle \quad \forall j = 1, ..., m \tag{3.5}$$

L'acquisizione può essere descritta in forma matriciale considerando la matrice Φ di dimensione $m\ x\ n$, avente per righe i trasposti dei vettori ϕ_j.

$$y = \Phi \cdot x \Rightarrow \begin{pmatrix} y_1 \\ ... \\ ... \\ ... \\ y_m \end{pmatrix} = \begin{pmatrix} [& \phi_1^T &] \\ & ... & \\ & ... & \\ & ... & \\ [& \phi_m^T &] \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ ... \\ ... \\ ... \\ ... \\ ... \\ x_n \end{pmatrix} \tag{3.6}$$

Questa equazione definisce anche il problema di ricostruzione del segnale. Come abbiamo visto all'inizio del capitolo, esistono infiniti segnali $u \in \Re^n$ che soddisfano $\Phi \cdot u = y$, ma sappiamo anche per ipotesi che x è un segnale sparso, quindi possiamo individuare la soluzione corretta imponendo un vincolo di sparsità sul segnale, dunque risolvendo il seguente problema di minimo:

$$\min_u \|u\|_0 \quad t.c. \quad \Phi u = \Phi x \tag{3.7}$$

con $m \geq 2k + 1$. Minimizzare la norma l_0 è però un problema NP-hard e quindi non applicabile direttamente ad un problema reale. È però stato dimostrato che la soluzione del problema (3.7) coincide con probabilità molto alta con quella del problema

$$\min_u \|u\|_1 \quad t.c. \quad \Phi u = \Phi x \tag{3.8}$$

se sono soddisfatte le seguenti condizioni:

1. la matrice di acquisizione Φ è di tipo casuale e gode della *Restricted Isometry Property* (RIP). Questo significa che ogni insieme di k colonne della matrice forma approssimativamente un sistema ortogonale, cioè:

$$1 - \delta_k \leq \frac{\|\Phi x\|_2}{\|x\|_2} \leq 1 + \delta_k \quad t.c. \quad \delta_k \in [0, 1] \quad \forall x \ \ k - sparso \tag{3.9}$$

2. m, il numero di misure lineari di x, soddisfa questo tipo di disuguaglianza:

$$m \geq c \cdot k \cdot log\left(\frac{n}{k}\right) \tag{3.10}$$

con c costante piccola. Questo limite in particolare vale per matrici gaussiane casuali. Il valore preciso del numero di misure lineari varierà a seconda della matrice di acquisizione considerata.

3.3 Applicazione della teoria alla ricostruzione di immagini

Se riconsideriamo il modello di acquisizione di un'immagine digitale (2.2), e lo confrontiamo con l'equazione (3.6), è evidente, come già anticipato, che la teoria del Compressed Sensing è applicabile anche nell'ambito delle immagini. La matrice di acquisizione sarà la matrice $H = D{\cdot}B$, di cui studieremo più approfonditamente le caratteristiche nel capitolo successivo. Per quanto riguarda la base di rappresentazione Ψ, esistono molteplici possibilità di scelta, ognuna delle quali permette diversi metodi per la risoluzione del problema di minimo.

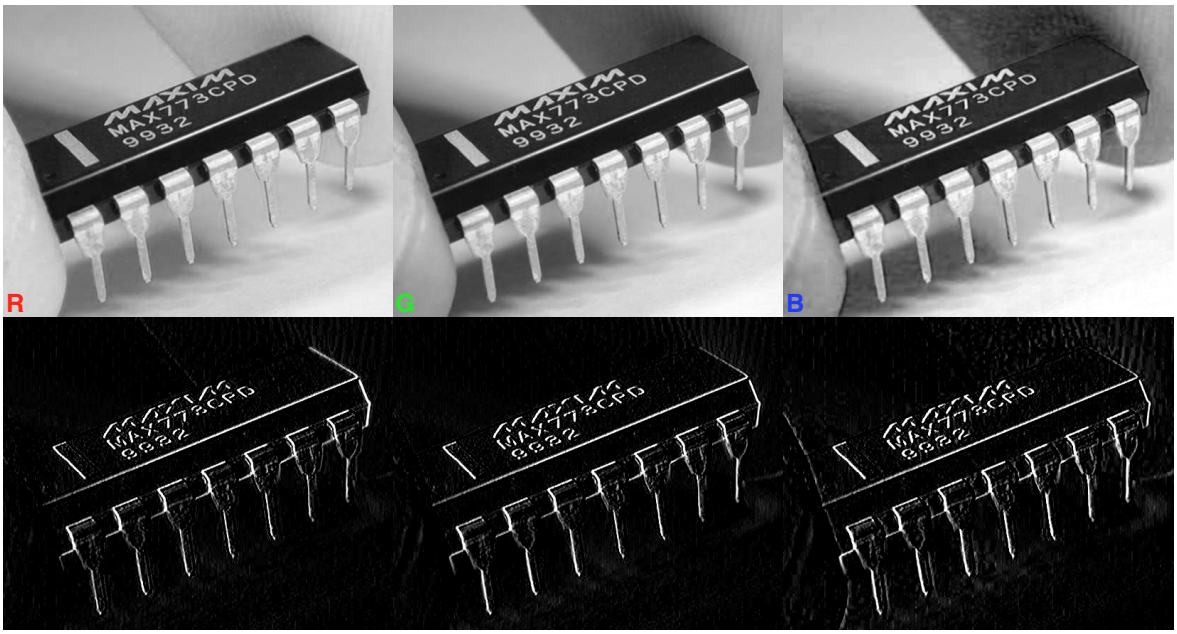


Fig 3.2: confronto fra i tre canali RGB di un'immagine e le rispettive rappresentazioni nel dominio del gradiente. È evidente come l'informazione sia sparsa e concentrata in corrispondenza dei bordi.

In questo lavoro prenderemo in considerazione un approccio particolarmente efficiente per questo tipo di problema: poiché le immagini possono essere modellate come funzioni regolari a tratti, con solo un piccolo numero di

discontinuità dovute ai bordi, il segnale può essere considerato sparso nel dominio del gradiente. Quindi possiamo riscrivere il problema (3.8) nella seguente forma:

$$\min_u \|\nabla u\|_1 \quad t.c. \quad \|Hu - g\|_2^2 \leq \epsilon^2 \tag{3.11}$$

il cui l'obiettivo è individuare il segnale u avente minima *Total Variation* e che contemporaneamente rispetti il vincolo di fedeltà ai dati. Il valore assegnato alla quantità ε^2 dipenderà dalla natura dell'immagine presa in esame: immagini con rappresentazione molto sparsa richiederanno valori prossimi allo zero, mentre immagini soltanto compressibili richiederanno valori più alti. Anche l'eventuale rumore presente nell'immagine influenzerà la scelta, rendendo necessaria una tolleranza più alta.

4 Algoritmo SR -CS

4.1 Metodo di risoluzione

Il problema (3.11) è particolarmente difficile da risolvere direttamente, in quanto il termine di regolarizzazione dato dal funzionale $J(u) = ||\nabla u||_1$, non è differenziabile. Allora una buona strategia è adottare un metodo di penalizzazione sostituendo il problema vincolato con una successione di sottoproblemi non vincolati

$$\min_{u\in\mathbb{R}^n} \left\{ J(u) + \frac{1}{2\lambda_k}\mathcal{H}(u) \right\} \tag{4.1}$$

con $\lambda_1 > \lambda_2 > \ldots > \lambda_{min}$, la cui soluzione converge a quella di (3.12) quando

$$\lim_{k\to\infty} \lambda_k = 0 \tag{4.2}$$

e dove

$$\mathcal{H}(u) = \|Hu - g\|_2^2 \tag{4.3}$$

rappresenta il termine di fedeltà all'immagine originale.

Questi problemi però risultano mal condizionati per valori piccoli del parametro di penalizzazione λ_k. Il mal condizionamento però può essere aggirato adottando un metodo di *splitting* che approssimi iterativamente la soluzione di ciascun sottoproblema. In [18] è stato introdotto un metodo di *proximal splitting* che risolve un problema del tipo

$$\min_{u\in\Re^n} f_1(x) + f_2(x) \tag{4.4}$$

in cui f_1 e f_2 sono due funzionali non convessi ed in particolare il primo non è differenziabile. Il principio è di utilizzare separatamente i due funzionali ad ogni iterazione, agendo con un passo *forward* esplicito su f_2 ed un passo *backward* implicito su f_1, che è coinvolto attraverso il suo operatore di prossimità. In

particolare l'operatore di prossimità del funzionale *J(u)* è la soluzione unica del problema di minimizzazione

$$Prox_{\lambda J}(v) = \min_{u \in \Re^n} \left\{ \lambda J(u) + \tfrac{1}{2}\|u - v\|_2^2 \right\} \tag{4.5}$$

Se il funzionale (4.3) è convesso, L-Lipschitziano e continuamente differenziabile, allora il sottoproblema (4.1) ammette almeno una soluzione caratterizzata dall'equazione del punto fisso:

$$u = Prox_{\beta\lambda J}(u - \beta\nabla\mathcal{H}(u)) \tag{4.6}$$

per ogni valore di β. Inoltre se $\beta \leq \dfrac{2}{L}$ dove L è il raggio spettrale della matrice H^TH

si ottiene il seguente schema iterativo *forward-backward* che permette di valutare la soluzione del problema di minimo (4.1):

$$u_{n+1} = Prox_{\beta\lambda J}(u_n - \beta\nabla\mathcal{H}(u_n)) \tag{4.7}$$

Quindi, per ciascun valore di λ_k, il sottoproblema (4.1) potrà essere risolto attraverso l'iterazione dei seguenti passi:

$$\begin{cases} v_{n+1} = u_n + \beta H^T(g - Hu_n) \\ u_{n+1} = Prox_{\beta\lambda_k J}(v_{n+1}) = \underset{u \in \Re^n}{\operatorname{argmin}} \left\{ \lambda_k J(u) + \tfrac{1}{2}\|u - v_{n+1}\|_2^2 \right\} \end{cases} \tag{4.8}$$

Dove il primo passo è l'aggiornamento del termine di fedeltà (passo forward), mentre il secondo è il passo di minimizzazione del termine di regolarizzazione (passo backward). Quest'ultimo in particolare può essere espresso per esteso come:

$$\tilde{u}_{n+1} = \underset{u \in \Re^n}{\operatorname{argmin}} \left\{ \lambda_k \|\nabla u\|_1 + \tfrac{1}{2}\|u - v_{n+1}\|_2^2 \right\} \tag{4.9}$$

e corrisponde alla formulazione di un passo di *denoising* sull'immagine perturbata v_{n+1} ottenuta dal precedente passo di aggiornamento.

Il metodo di *proximal forward-backward splitting* appartiene ad una classe di algoritmi nota come ISTA (*Iterative Shrinkage-Thresholding Algorithms*) che hanno la caratteristica di essere molto semplici, ma con lo svantaggio di avere una

convergenza alla soluzione molto lenta. Per rimediare a questo inconveniente in letteratura sono stati studiati diversi metodi alternativi che, anziché calcolare l'iterato successivo basandosi unicamente su quello dell'ultima iterazione, sfruttano due o più iterati già calcolati nei passi precedenti. In particolare in [19] è stato introdotto FISTA (*Fast Iterative Shrinkage-Thresholding Algorithm*) che riesce a migliorare la velocità di convergenza del metodo, mantenendone al contempo la semplicità. Nella pratica ai passi già visti in (4.8) viene aggiunta una nuova regola per l'aggiornamento della soluzione:

$$\begin{cases} v_{n+1} = u_n + \beta H^T(g - Hu_n) \\ \tilde{u}_{n+1} = Prox_{\beta\lambda_k J}(v_{n+1}) = \underset{u\in\Re^n}{\operatorname{argmin}} \left\{ \lambda_k J(u) + \frac{1}{2}\|u - v_{n+1}\|_2^2 \right\} \\ u_{n+1} = \tilde{u}_{n+1} + \tau_n(\tilde{u}_{n+1} - \tilde{u}_n) \qquad t.c. \qquad \tau_n = \frac{t_n - 1}{t_{n+1}} \end{cases} \tag{4.10}$$

dove i valori di t sono dati dalla sequenza

$$t_1 = 1 \quad e \quad t_{n+1} = \frac{1 + \sqrt{1 + 4t_n^2}}{2} \tag{4.11}$$

Una volta definito il metodo di risoluzione dei sottoproblemi è necessario stabilire una regola per la riduzione del parametro λk. Nell'ambito del Compressed Sensing viene in genere adottata la seguente formula:

$$\lambda_{k+1} = \lambda_k \cdot r \quad t.c. \quad r \in (0, 1) \tag{4.12}$$

dove di solito viene scelto $r = 0.5$. Lo svantaggio di questa regola è che la scelta di un valore arbitrario per il parametro r potrebbe impedire al metodo di raggiungere un minimo corrispondente ad una buona soluzione. Un criterio di aggiornamento migliore proposto in [20] è ottenuto tenendo in considerazione l'evoluzione del problema, attraverso il funzionale

$$F(\lambda_k, u_{\lambda_k}) = \frac{1}{2}\mathcal{H}(u_{\lambda_k}) + \lambda_k J(u_{\lambda_k}) \tag{4.13}$$

dove u_{λ_k} è la soluzione del sottoproblema precedente:

$$u_{\lambda_k} = \underset{u\in\Re^n}{\operatorname{argmin}} \left\{ \frac{1}{2}\mathcal{H}(u) + \lambda_k J(u) \right\} \tag{4.14}$$

Il nuovo criterio di aggiornamento sarà dunque:

$$\lambda_{k+1} = \lambda_k \cdot \frac{F(\lambda_k, u_{\lambda_k})}{F(\lambda_{k-1}, u_{\lambda_{k-1}})} \tag{4.15}$$

Per essere sicuri dell'efficacia di questa nuova regola è necessario verificare che quella generata sia una sequenza monotona decrescente. Consideriamo dunque due possibili valori del parametro di penalizzazione λ_{k-1} e λ_k tali che $\lambda_{k-1} > \lambda_k$ e siano $u_{\lambda k-1}$ e $u_{\lambda k}$ le relative minimizzazioni di (4.14). Si dimostra che

$$F(\lambda_{k-1}, u_{\lambda_{k-1}}) > F(\lambda_k, u_{\lambda_k}), \tag{4.16}$$

quindi la (4.15) genererà di sicuro una sequenza decrescente di valori. Inoltre poiché

$$\begin{aligned} \lambda_{k+1} &= \lambda_k \cdot \frac{F(\lambda_k, u_{\lambda_k})}{F(\lambda_{k-1}, u_{\lambda_{k-1}})} = \lambda_{k-1} \cdot \frac{F(\lambda_k, u_{\lambda_k})}{F(\lambda_{k-2}, u_{\lambda_{k-2}})} = \ ... \\ ... \ &= \frac{\lambda_2}{F(\lambda_1, u_{\lambda_1})} \cdot F(\lambda_k, u_{\lambda_k}) \end{aligned} \tag{4.17}$$

è evidente come il valore di λ_{k+1} dipenda in realtà solo dai primi due valori $\lambda_1 > \lambda_2$ e poiché questi non sono definiti in maniera formale, ma piuttosto in base a prove sperimentali, è possibile riformulare il criterio di aggiornamento come:

$$\lambda_{k+1} = Const \cdot F(\lambda_k, u_{\lambda_k}) \tag{4.18}$$

L'ultimo dettaglio necessario per la stesura dell'algoritmo è la definizione del valore iniziale di λ attraverso la seguente formula:

$$\lambda_1 = \frac{\mathcal{H}(u_0)}{2J(u_0)} \cdot w \quad t.c. \quad w > 0 \tag{4.19}$$

che bilancia i fattori di fedeltà all'immagine originale e la regolarizzazione.

L'algoritmo così ottenuto è presentato nella sua interezza nella pagina seguente.

Algoritmo SR-CS

Dati g, H, w, β, out-it, in-it

Inizializzazione

$$u_0 = g, \quad \tilde{u}_0 = u_0, \quad \lambda_1 = \frac{\mathcal{H}(u_0)}{2J(u_0)} \cdot w$$

$$v_1 = u_0 + \beta H^T(g - Hu_0)$$

$$u_1 = \underset{u \in \Re^n}{\operatorname{argmin}} \left\{ \lambda_1 J(u) + \tfrac{1}{2}\|u - v_1\|_2^2 \right\}, \quad \tilde{u}_1 = u_1$$

$$\lambda_2 = \lambda_1 \cdot \frac{F(\lambda_1, u_1)}{F(\lambda_1, u_0)}$$

if $\lambda_2 = \lambda_1$

then $\lambda_2 = \lambda_1 \cdot 0.5$

for $k = 2$: out-it

$t_0 = 1$

for $n = 1$: in-it

$$v_{n+1} = u_n + \beta H^T(g - Hu_n)$$

$$\tilde{u}_{n+1} = \underset{u \in \Re^n}{\operatorname{argmin}} \left\{ \lambda_k J(u) + \tfrac{1}{2}\|u - v_{n+1}\|_2^2 \right\}$$

$$t_{n+1} = \frac{1 + \sqrt{1 + 4t_n^2}}{2}, \quad \tau_n = \frac{t_n - 1}{t_{n+1}}$$

$$u_{n+1,\lambda_k} = u_{n+1,\lambda_k} + \tau_n(u_{n+1,\lambda_k} - \tilde{u}_{n,\lambda_k})$$

end

$$F(\lambda_k, u_{n+1,\lambda_k}) = \frac{1}{2}\mathcal{H}(u_{n+1,\lambda_k}) + \lambda_k J(u_{n+1,\lambda_k})$$

$$\lambda_{k+1} = Const \cdot F(\lambda_k, u_{\lambda_k})$$

end

4.2 Scelta della matrice H

L'algoritmo per essere efficace presuppone però la conoscenza della matrice di acquisizione H. Questa è già stata introdotta nel secondo capitolo ed è composta da una fase di blurring ed una successiva fase di decimazione, descritte rispettivamente dalle matrici B e D. Inoltre è necessario che la matrice H goda della RIP.

Se determinare la matrice D non costituisce una grande difficoltà, può infatti essere generata automaticamente in base al fattore di upscaling che si desidera ottenere, il problema principale è invece proprio individuare la matrice di blur.

$$\begin{bmatrix}1\\3\\5\\7\end{bmatrix} = \begin{bmatrix}1&0&0&0&0&0&0&0\\0&0&1&0&0&0&0&0\\0&0&0&0&1&0&0&0\\0&0&0&0&0&0&1&0\end{bmatrix} \cdot \begin{bmatrix}1\\2\\3\\4\\5\\6\\7\\8\end{bmatrix}$$

$$\begin{bmatrix}1\\5\end{bmatrix} = \begin{bmatrix}1&0&0&0&0&0&0&0\\0&0&0&0&1&0&0&0\end{bmatrix} \cdot \begin{bmatrix}1\\2\\3\\4\\5\\6\\7\\8\end{bmatrix}$$

Fig 4.1: un esempio di matrici di decimazione che riducono un vettore rispettivamente di un fattore 2 e di un fattore 4.

Ovviamente la soluzione ideale sarebbe riuscire a ricostruire la matrice B in modo che sia il più fedele possibile a quella applicata in fase di acquisizione. Purtroppo però non abbiamo alcun controllo su questa fase e l'unico input a nostra disposizione in questo problema è l'immagine iniziale a bassa risoluzione. Inoltre, come è stato illustrato nel primo capitolo, durante l'acquisizione intervengono numerosi fattori di sfocatura che rendono la matrice B estremamente variabile, non solo fra differenti dispositivi di acquisizione, ma addirittura da acquisizione a acquisizione. Scartata quindi l'ipotesi di ricostruire fedelmente la matrice di blur, l'opzione migliore è individuare un modello sufficientemente flessibile da funzionare bene nel maggior numero di applicazioni. Fortunatamente in [21] è stato dimostrato che matrici di *Toeplitz* generate casualmente godono della RIP. Poiché i filtri di blur sono in genere implementati proprio utilizzando delle matrici di Toeplitz, c'è un'alta probabilità che la matrice H goda della RIP indipendentemente dal modello di blur scelto.

In [5] è stato proposto l'utilizzo di un filtro gaussiano con media zero e deviazione standard σ compresa fra *1* e *2*. Questi filtri sono sufficientemente versatili e permettono di ottenere dei buoni risultati, privi di sfocature o scalettature eccessive sulle immagini, a patto però di impostare accuratamente una serie di parametri fra cui, oltre allo stesso σ, anche le dimensioni del filtro. Questo va a scapito dell'immediatezza dell'algoritmo che richiede sempre l'intervento dell'utente per avere di volta in volta la calibrazione più idonea all'immagine presa in esame. Esistono anche dei metodi per stimare la deviazione standard del blurring gaussiano applicato ad un'immagine, ma questi hanno o costi computazionali molto elevati, come [22] basato sulla trasformata di Radon, o richiedono un database di immagini sufficientemente ampio per addestrare il metodo, come [23]. Inoltre, benché siano in grado di individuare con una certa precisione una stima globale del blur applicato ad un'immagine, non sono adatti al nostro scopo: infatti è impossibile rilevare il blur generato dal sottocampionamento del sensore partendo da un'immagine nella sua risoluzione "naturale".

L'idea proposta in questo lavoro è di adottare i filtri utilizzati nella decomposizione wavelet per modellare la matrice B. Questa scelta è stata suggerita dalla natura stessa di questi filtri, intrinsecamente legati al concetto di abbassamento di risoluzione. Se infatti consideriamo un'immagine di dimensioni NxN, descritta dalla matrice C_0, la sua decomposizione wavelet è ottenuta con la seguente formula:

$$\begin{pmatrix} C_{-1} & D^1_{-1} \\ D^2_{-1} & D^3_{-1} \end{pmatrix} = \begin{pmatrix} \begin{bmatrix} H_y \end{bmatrix} \\ \begin{bmatrix} W_y \end{bmatrix} \end{pmatrix} \cdot \begin{pmatrix} C_0 \end{pmatrix} \cdot \left(\begin{bmatrix} H_x^T \end{bmatrix} \begin{bmatrix} W_x^T \end{bmatrix} \right)$$

dove H_x, H_y, W_x e W_y sono tutte matrici di dimensioni $\frac{N}{2} x N$ e combinano un'operazione di decimazione, che dimezza il numero delle informazioni, con l'applicazione di un filtro: passa basso nel caso delle prime due e passa alto per le altre. Il risultato della decomposizione è una nuova matrice composta da quattro porzioni, ciascuna di dimensione $\frac{N}{2} x \frac{N}{2}$, delle quali in particolare C_{-1} isola le basse

frequenze dell'originale, mentre le restanti riportano le alte frequenze. In particolare il dimezzamento delle dimensioni è giustificato dallo stesso teorema di Nyquist-Shannon, in quanto l'ampiezza di banda ridotta richiede un numero di campioni minore.

Fig 4.2: illustrazione della decomposizione wavelet di un'immagine. L'applicazione successiva degli stessi filtri permette ottenere una versione a bassa risoluzione dell'immagine.

L'immagine C_{-1} può essere ulteriormente sottoposta ad un processo di decomposizione utilizzando gli stessi filtri opportunamente dimensionati. Naturalmente per i nostri scopi è sufficiente tenere in considerazione solo i filtri passa basso, infatti la porzione di immagine relativa alle basse frequenze è a tutti gli effetti una riduzione dell'immagine originale.

4.3 Implementazione

L'algoritmo per la super-resolution illustrato in questo capitolo è stato implementato in Matlab. Preservare la notazione lessicografica delle immagini avrebbe potuto rivelarsi problematico, infatti le matrici coinvolte hanno dimensioni notevoli: presa un'immagine di dimensioni NxN, le relative matrici B e D avrebbero rispettivamente dimensioni N^2xN^2 e $\left(\frac{N}{d}\right)^2 xN^2$, quindi anche immagini con un modesto numero di pixel richiederebbero una grande quantità di memoria. Dunque sono state considerate le matrici B_x, B_y, D_x e D_y il cui prodotto tensoriale è:

$$B_x \otimes B_y = B \quad e \quad D_x \otimes D_y = D$$

quindi il processo di acquisizione $g = D{\cdot}B{\cdot}f$ è stato sostituito con la seguente formulazione equivalente:

$$g = (D_x \otimes D_y) \cdot (B_x \otimes B_y) \cdot f \;\Rightarrow\; G = D_y \cdot B_y \cdot F \cdot B_x^T \cdot D_x^T$$

Le matrici di blur B_x e B_y sono costruite a partire dai filtri passa basso *Daubechies* (di seguito DB) con *4 vanishing moment* (DB4), composti da una serie di valori h_i per $i = 1,...,8$. Come anticipato sono matrici di Toeplitz, quindi caratterizzate dalla seguente struttura:

$$\begin{pmatrix} ... & ... & ... & ... & ... & ... & ... & ... & ... & ... \\ ... & h_0 & h_1 & ... & h_8 & 0 & 0 & 0 & 0 & ... \\ ... & 0 & h_0 & h_1 & ... & h_8 & 0 & 0 & 0 & ... \\ ... & 0 & 0 & h_0 & h_1 & ... & h_8 & 0 & 0 & ... \\ ... & ... & ... & ... & ... & ... & ... & ... & ... & ... \\ ... & 0 & 0 & 0 & 0 & h_0 & h_1 & ... & h_8 & ... \\ ... & ... & ... & ... & ... & ... & ... & ... & ... & ... \end{pmatrix}$$

Per evitare la formazione di artefatti in prossimità dei bordi, si è deciso di estendere le immagini applicando un padding, che viene poi rimosso dal risultato finale. Le matrici così costruite però possono essere utilizzate soltanto per un downsample di fattore *2*, mentre altri fattori di sottocampionamento, come abbiamo visto, possono essere ottenuti attraverso applicazioni successive degli stessi filtri. Ad esempio una riduzione di fattore *4* può essere ricavata con la seguente operazione:

$$G = D_{2y} \cdot B_{2y} \cdot D_{1y} \cdot B_{1y} \cdot F \cdot B_{1x}^T \cdot D_{1x}^T \cdot B_{2x}^T \cdot D_{2x}^T$$

Per mantenere un ridotto numero di moltiplicazioni per passi di upscaling d superiori a *2* fra matrici, si è deciso di implementare il filtri nel seguente modo:

1. viene generato un nuovo vettore intervallando gli elementi del filtro DB con $\frac{d}{2} - 1$ elementi nulli;
2. il filtro utilizzato per il passo di upscaling $\frac{d}{2}$ viene invertito (*flip*);
3. viene calcolato un nuovo filtro come risultato della convoluzione dei vettori dei due punti precedenti;

4. il vettore così ottenuto viene utilizzato per generare la matrice di Toeplitz B_x (o B_y).

Infine le matrici di blur vengono normalizzate dividendole per $\sqrt{d}$. Questo previene la perdita di energia del segnale, che altrimenti risulterebbe in un'immagine con ridotta luminosità.

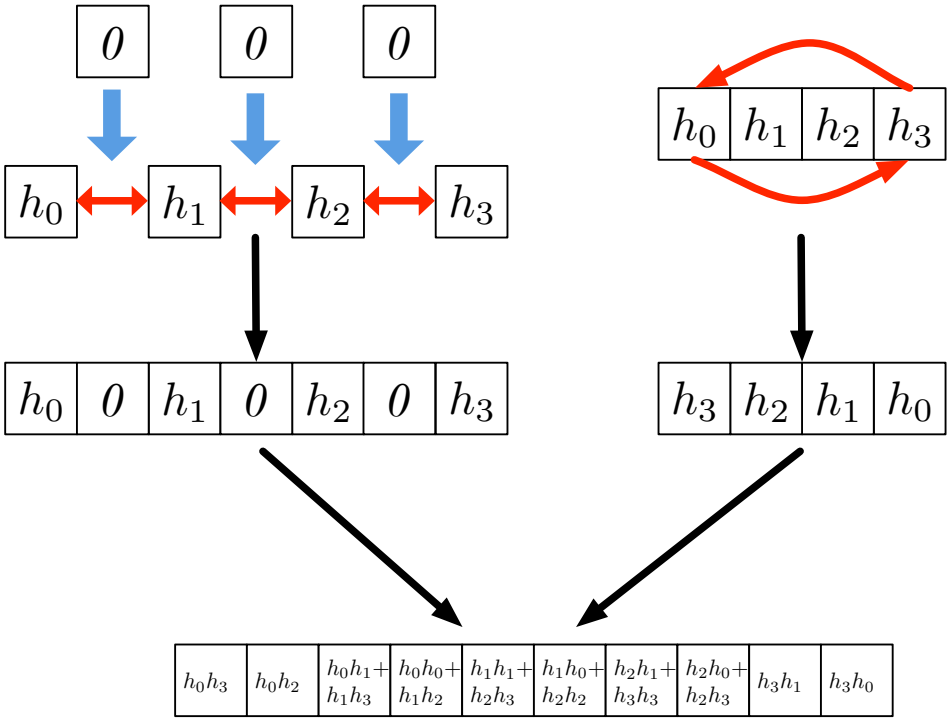


Fig 4.3: esempio di costruzione di un filtro di blur per upscaling di fattore d = 4. In questo caso sono stati utilizzati filtri DB con 2 vanishing moment.

L'approccio a molti livelli di risoluzione appena descritto non vincola alla scelta di un particolare tipo di filtro, ma può essere applicato anche a filtri diversi da quelli tipicamente utilizzati nella decomposizione wavelet. Si è quindi deciso di implementare anche una versione *multi-resolution* dei filtri gaussiani utilizzati in [5]. Di fatto, il filtro ottenuto attraverso la convoluzione sarà un nuovo filtro gaussiano con dimensioni e σ differenti. L'algoritmo resterà quindi efficace, con il vantaggio di essere semplificato per l'utente: le dimensioni e la costante σ corrette per un passo di upscaling 2 vengono infatti automaticamente ridimensionate per gli altri fattori.

Per ridurre ulteriormente il numero dei parametri definiti dall'utente, è stata definiti una regola di arresto per le iterazioni interne dell'algoritmo. In pratica le iterazioni interne vengono ripetute finché vale la condizione:

$$\frac{|F(\lambda_k, u_{n,\lambda_k}) - F(\lambda_k, u_{n-1,\lambda_k})|}{F(\lambda_k, u_{n-1,\lambda_k})} > \gamma \cdot \lambda_k$$

dove γ è un parametro definito da utente.Il vantaggio di questo criterio è di applicare di volta in volta il numero di iterazioni più adeguato, troncando nel caso non ci siano miglioramenti nella soluzione o continuando ad iterare fino a raggiungere una soluzione migliore.

Ulteriori accorgimenti adottati nell'implementazione sono l'utilizzo dell'algoritmo di Chambolle con *5* (o *7*) iterazioni per il passo backward dell'algoritmo e l'anticipazione dei passi di risoluzione interni nelle iterazioni esterne.

5 Sperimentazione

5.1 Impostazione dei test

Per testare e comprendere al meglio le caratteristiche e le capacità dell'algoritmo presentato in questa tesi, è stata approntata una serie di prove i cui risultati sono presentati in questo capitolo. Per le sperimentazioni sono state utilizzate immagini di test di varia natura, sia per quanto riguarda i soggetti inquadrati, sia per quanto riguarda la tipologia di acquisizione. In particolare per le immagini a colori l'algoritmo è stato applicato separatamente ai canali delle tre componenti RGB, utilizzando sempre gli stessi parametri. Per facilitare l'interpretazione dei risultati, questi verrano confrontati con le immagini ad alta risoluzione prodotte da alcuni degli algoritmi presentati nel secondo capitolo. In particolare sono stati selezionati i seguenti metodi:

- interpolazione bicubica, in quanto uno dei metodi più diffusi e presente nei più comuni programmi di elaborazione grafica;
- NEDI, perché, nonostante siano stati sviluppati numerosi approcci con risultati migliori, costituisce uno dei riferimenti più utilizzati in letteratura;
- ICBI, perché è un metodo di interpolazione migliore del precedente, ma comunque basato sull'orientazione degli edge.

Oltre a questi, l'algoritmo è stato confrontato anche con l'implementazione precedente di SR-CS, basata sull'utilizzo di filtri gaussiani. I parametri utilizzati per questa versione sono quelli presentati in [5]. Tutti gli algoritmi considerati sono implementati in Matlab.

Nonostante il principale metodo di valutazione dei risultati sia il confronto visivo diretto, si è deciso di adottare anche alcuni criteri di valutazione oggettiva, in particolare le misure scelte sono:

- PSNR (*Peak Signal to Noise Ratio*), utilizzato comunemente per valutare la qualità degli algoritmi di compressione e calcolato come:

$$PSNR = 20 \cdot \log_{10} \left(\frac{max\{F\}}{RMSE} \right)$$

dove F indica l'immagine originale considerata ed RMSE l'errore quadratico medio così ottenuto:

$$RMSE = \sqrt{\frac{\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}[F_{i,j} - U_{i,j}]^2}{m \cdot n}}$$

ossia la differenza fra il segnale originale e quello ricostruito. Immagini di qualità migliore saranno caratterizzate da RMSE basso e PSNR alto;

- PEE (*Percentage Edge Error*), che stima la bontà della ricostruzione degli edge nell'immagine ingrandita U ed è calcolato in questo modo:

$$PEE = \frac{\|\nabla F\|_2 - \|\nabla U\|_2}{\|\nabla F\|_2} \cdot 100,$$

un valore positivo significherà che l'immagine ricostruita presenta un eccessivo smoothing, mentre un valore negativo, che sono presenti degli edge spuri [24].

Queste due misure non sono però in grado di restituire sempre un valore affidabile, infatti la definizione di una metrica precisa per la valutazione degli algoritmi di upscaling è un problema tuttora aperto. Questo rende di fatto imprescindibile la valutazione visiva. Inoltre per poter sfruttare queste misure è necessario disporre anche delle immagini originali ad alta risoluzione ed impostare il test come se fosse un ripristino della risoluzione. Purtroppo non esistono database di riferimento per questo problema, è stato quindi necessario reperire alcune immagini ad alta risoluzione e simulare il processo di acquisizione ad un livello di risoluzione più basso. Per farlo si è deciso di adottare il metodo di interpolazione bicubica implementato in GIMP (*GNU Image Manipulation Program*), in modo da non avere alcun controllo su questa fase.

Nessuno però garantisce che il metodo di interpolazione bicubica sia un buon modello per l'acquisizione di immagini a bassa risoluzione. Si è quindi deciso di produrre delle ulteriori immagini di test utilizzando un dispositivo di acquisizione dalle capacità limitate, dotato di un sensore da *0,3* megapixel (*640x480* pixel). Le immagini così ottenute non permettono la valutazione numerica dei risultati, ma indubbiamente offrono l'occasione di confrontare i vari algoritmi su casi verosimili, in quanto la riduzione di risoluzione non è introdotta artificialmente utilizzando un qualche algoritmo.

5.2 Impostazione dei parametri

Nonostante la scelta dell'approccio multi-resolution e dei filtri DB semplifichi l'impostazione dei parametri, il metodo SR-CS, per poter funzionare in maniera ottimale, necessita comunque di una calibrazione iniziale di alcune variabili.
I parametri da impostare sono:

- il peso w utilizzato nell'inizializzazione di λ;
- il valore $Const$ utilizzato per l'aggiornamento di λ;
- il numero di iterazioni esterne out;
- il valore di γ utilizzato per l'arresto automatico delle iterazioni interne;
- il numero massimo di iterazioni interne in_max.

Il valore di β in base alle sperimentazioni condotte ed ai dati presentati in [5] è stato fissato a 1.

Il peso w contribuisce a definire il valore iniziale del parametro λ. Utilizzare un peso maggiore significherà partire con un valore iniziale di λ molto grande rallentando la velocità di convergenza. Valori più piccoli oltre a velocizzare il metodo, aumenteranno il contrasto delle immagini. Nelle sperimentazioni sono stati utilizzati valori compresi fra $0,01$ e $0,1$.

Fig. 5.1: influenza del peso nella ricostruzione. A) w = 0,1; Const = 0,00005; γ= 0,01; out = 4; in_max = 20; tempo = 136,5945 secondi. B) w = 0,01; Const = 0,00005; γ= 0,01; out = 4; in_max = 20; tempo = 135,7365 secondi.

$Const$ influenza l'aggiornamento di λ e si comporta in modo analogo: un valore piccolo permette di mantenere più dettagli seppur con una maggior scalettatura.
Il range adottato nei test va da $0,000005$ a $0,00005$.

Fig. 5.2: influenza della costante nella ricostruzione. A) w = 0,1; Const = 0,00005; γ= 0,01; out = 4; in_max = 20; tempo = 136,5945 secondi. B) w = 0,1; Const = 0,000005; γ= 0,01; out = 4; in_max = 20; tempo = 136,75 secondi.

Il numero di iterazioni esterne definisce il numero di sottoproblemi in cui è scomposto il problema di minimizzazione iniziale. Un numero piccolo produrrà un risultato molto sfuocato, mentre un numero grande porterà ad una maggior regolarizzazione e quindi ad un contrasto maggiore. In genere *3* o *4* iterazioni esterne costituiscono una scelta ottimale.

Fig. 5.3: influenza delle iterazioni esterne nella ricostruzione. A) w = 0,1; Const = 0,00005; γ= 0,01; out = 4; in_max = 20; tempo = 136,5945 secondi. B) w = 0,1; Const = 0,00005; γ= 0,01; out = 6; in_max = 20; tempo = 205,5469 secondi.

Ognuno dei sottoproblemi corrispondenti alle iterazioni esterne viene a sua volta risolto iterativamente. Il parametro γ stabilisce con quale velocità le iterazioni

interne vengono arrestate. Assegnare un valore alto significa ridurre il numero delle iterazioni e quindi la sfocatura delle immagini.

Fig. 5.4: influenza di γ sulla ricostruzione. A) w = 0,1; Const = 0,00005; γ= 0,01; out = 4; in_max = 20; tempo = 136,5945 secondi. B) w = 0,1; Const = 0,00005; γ= 0,1; out = 4; in_max = 20; tempo = 88,6398 secondi.

Una caratteristica di γ è di agire in maniera differente a seconda dell'iterazione esterna considerata: a parità di valore tende a troncare più velocemente le iterazioni interne in corrispondenza delle prime iterazioni esterne. Per riuscire ad ottenere un risultato gradevole e non affetto da eccessiva scalettatura è necessario che la prima iterazione esterna abbia un buon numero di iterazioni interne, dunque è consigliabile utilizzare un valore abbastanza piccolo. Nei test sono stati utilizzati valori compresi fra $0,01$ e $0,1$.

Purtroppo un valore alto di γ significa anche un numero elevato di iterazioni interne nelle fasi più avanzate di esecuzione dell'algoritmo, con notevoli ricadute sul tempo di esecuzione e spesso anche sull'efficacia del metodo stesso: valori alti per i parametri w e $Const$ rendono il metodo suscettibile ad un numero troppo elevato di iterazioni interne. Si è quindi deciso di inserire un limite massimo per queste iterazioni, che per gli esperimenti è stato fissato a 20, ma è possibile utilizzare valori maggiori nel caso si adottino w e $Const$ alti.

Fig. 5.5: influenza del numero massimo di iterazioni interne sulla ricostruzione. A) w = 0,1; Const = 0,00005; γ= 0,01; out = 4; in_max = 20; tempo = 136,5945 secondi. B) w = 0,01; Const = 0,00001; γ= 0,01; out = 4; in_max = 30; tempo = 199,9465 secondi.

Purtroppo il metodo non funziona bene a prescindere dall'immagine in input. Come si può notare in figura 5.6, i parametri che si rivelano validi per un'immagine, potrebbero essere inadeguati ad un'immagine pressoché identica, ma acquisita in maniera differente.

Fig. 5.6: entrambe le immagini sono state ottenute con i seguenti parametri: w = 0,1; Const = 0,00001; γ= 0,01; out = 4; in_max = 20. A) ridotta utilizzando l'interpolazione bicubica di GIMP presenta evidenti scalettature. B) ridotta con l'interpolazione bicubica di Matlab ha bordi più lisci.

Alla fine del capitolo precedente è stata brevemente introdotta anche l'implementazione dell'algoritmo basata su filtri di blurring gaussiano multi-resolution. Questo metodo condivide gran parte dei parametri finora descritti per

l'implementazione con filtri DB, ma date le difficoltà iniziali nella configurazione, alcuni di questi sono stati disabilitati. In particolare non vengono utilizzati i parametri $Const$ e γ, in quanto si è optato per una regola di aggiornamento a passo fisso per il parametro λ e per un numero di iterazioni interne fissato per ciascuna delle esterne, in modo che esse decrescano durante l'esecuzione del metodo. Questa implementazione richiede ovviamente la definizione delle dimensioni del filtro gaussiano ($filt\_size$) e del σ: filtri di grandi dimensioni corrispondenti a σ grande produrranno risultati più sfocati, mentre filtri corti corrispondenti a σ piccolo daranno risultati con maggior contrasto [5].

5.3 Ingrandimento di fattore 4

La prima immagine "*Lena"* è caratterizzata da molti dettagli (alte frequenze): i capelli, le piume ed il tessuto del cappello; è dunque un'immagine molto ostica per questo genere di problema. Per testare il ripristino di risoluzione l'immagine originale di dimensioni $512x512$ pixel è stata ridotta con GIMP a $128x128$ pixel. In figura 5.7 è presentato il confronto fra l'immagine originale ad alta risoluzione e l'immagine ridotta utilizzata come input. I metodi presentati in questa tesi sono stati testati variando l'inizializzazione dei parametri e fra tutti i risultati ottenuti vengono presentati quelli ritenuti migliori. In figura 5.8 è presentato il confronto fra le immagini ottenute, mentre la tabella 1 riporta i risultati ottenuti dagli algoritmi.

Fig. 5.7: "Lena", confronto fra l'immagine originale 512x512 pixel e la sua versione ridotta a 128x128 pixel.

SR-CS DB4 *SR-CS Multigauss*

Interpolazione bicubica *NEDI*

ICBI *SR-CS Gauss*

Fig. 5.8: ingrandimento fattore 4. Parametri DB4 w = 0,1; Const = 0,00005; γ = 0,05; out = 4; in_max = 20. Parametri Multigauss: w = 1/80; filt_size = 7; σ = 1; out = 3.

Metodo	PSNR	PEE	Tempo (sec.)
SR-CS DB4	33.265513	12.062168	95.953125
SR-CS Multigauss	33.183986	-10.475971	76.828125
Bicubica	35.106789	11.928146	0.078125
NEDI	33.156219	12.274021	53.984375
ICBI	34.597199	8.769312	87.671875
SR-CS Gauss	33.272592	-7.448051	52.281250

Tabella 1: risultati dei test relativi al ripristino di risoluzione di "Lena" da 128x128 a 512x512 pixel.

I metodi proposti hanno risultati paragonabili agli algoritmi di confronto. Sono caratterizzati da un buon contrasto, in particolare Multigauss, e non presentano particolari artefatti. Il metodo utilizzato per il calo di risoluzione introduce una scalettatura nell'immagine che però nessuno degli algoritmi riesce ad evitare completamente. Per quanto riguarda i tempi, l'implementazione con filtri DB impiega più tempo rispetto gli altri algoritmi, mentre la versione con filtri gaussiani multi-resolution, date anche le semplificazioni nell'implementazione, ha tempi paragonabili agli altri metodi. In figura 5.9 sono presentati i dettagli delle immagini in scala 1 a 1 per facilitare il confronto.

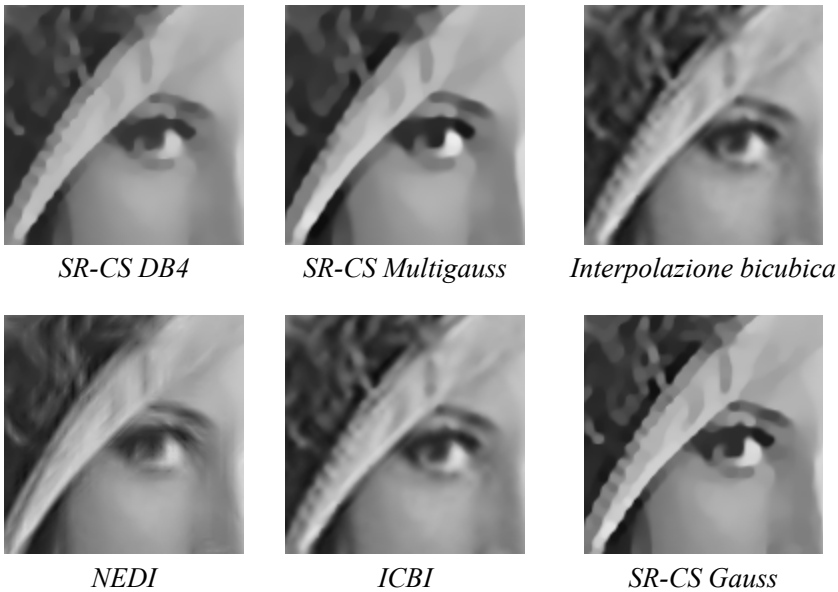


Fig. 5.9: confronto a dimensioni reali fra dettagli (120x120 pixel) degli ingrandimenti di fattore 4 presentati nella figura precedente.

Il secondo test consiste in un altro ripristino di risoluzione, l'immagine originale *"Bambino"* è stata ridotta da *512x512* pixel a *128x128* pixel utilizzando GIMP.

Anche questa immagine presenta numerosi dettagli (il cappello di lana, le ciglia), ma anche delle imperfezioni legate all'acquisizione originale: si intravedono delle righe sul volto, probabilmente dovute ad una scansione dalla pagina di una rivista. In figura 5.10 è presentato un confronto fra l'immagine originale e la sua versione a bassa risoluzione.

Fig. 5.10: "Bambino", confronto fra l'immagine originale 512x512 pixel e la sua versione ridotta a 128x128 pixel.

In figura 5.11 è presentato il confronto fra le immagini ottenute, mentre la tabella 2 riporta i risultati ottenuti dai vari algoritmi.

Metodo	PSNR	PEE	Tempo (sec.)
SR-CS DB4	33.200544	1.969334	309.781250
SR-CS Multigauss	32.452257	-16.695118	228.500000
Bicubica	35.113235	5.635672	0.046875
NEDI	32.971694	6.189321	160.625000
ICBI	34.8256	3.8595	223.125000
SR-CS Gauss	33.007939	-6.046628	155.125000

Tabella 2: risultati dei test relativi al ripristino di risoluzione di "Bambino" da 128x128 a 512x512 pixel.

SR-CS DB4 *SR-CS Multigauss*

Interpolazione bicubica *NEDI*

ICBI *SR-CS Gauss*

Fig. 5.11: ingrandimento fattore 4. Parametri DB4 w = 0,05; Const = 0,00001; γ = 0,1; out = 4; in_max = 20. Parametri Multigauss: w = 1/80; filt_size = 5; σ = 1,5; out = 3.

Per quanto riguarda il PSNR, le due implementazioni hanno risultati paragonabili ai metodi di confronto, sebbene non riescano a raggiungere il valore più alto relativo all'interpolazione bicubica (avvantaggiata per il fatto di essere basata sullo stesso metodo utilizzato per la riduzione di risoluzione). Il metodo con filtri DB4 riesce ad ottenere il PEE più vicino allo zero, mentre il metodo con filtri gaussiani multi-resolution, non riesce a ricostruire fedelmente tutti i dettagli, ma in compenso è in grado di eliminare la scalettature dei bordi e di offrire contemporaneamente un buon contrasto. Entrambi i metodi presentati migliorano la tessitura dell'immagine, rimuovendo il difetto delle righe sul volto. In figura 5.12 sono presentati i dettagli delle immagini in scala 1 a 1 per facilitare il confronto.

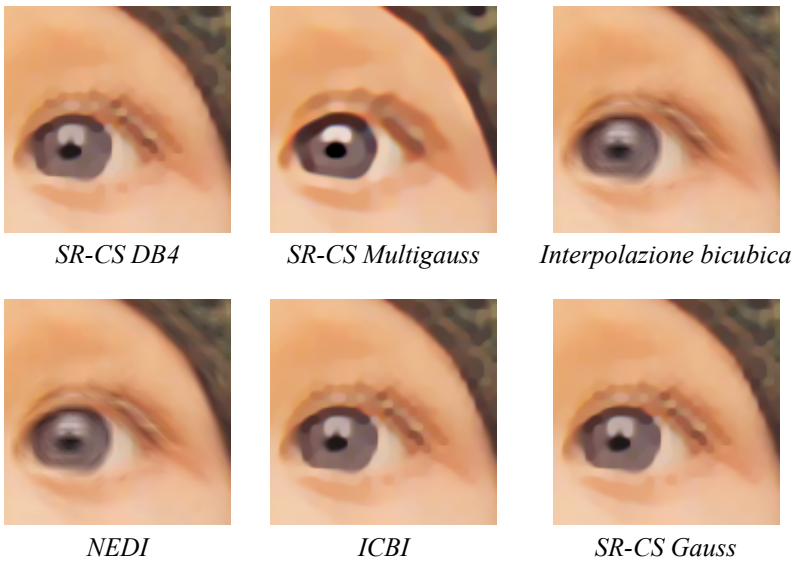
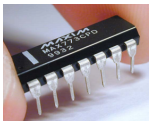
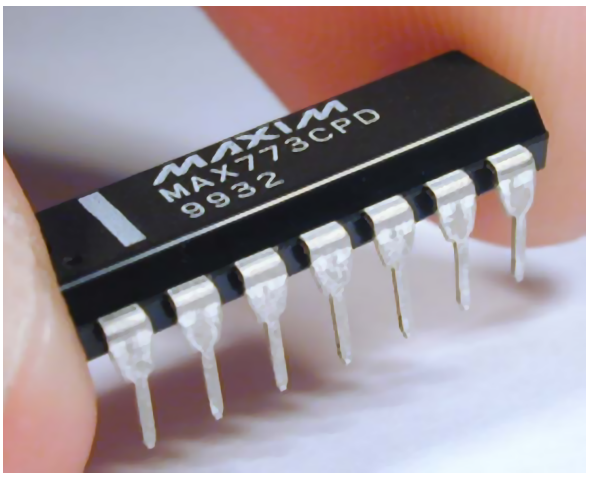


Fig. 5.12: confronto a dimensioni reali fra dettagli (120x120 pixel) degli ingrandimenti di fattore 4 presentati nella figura precedente.

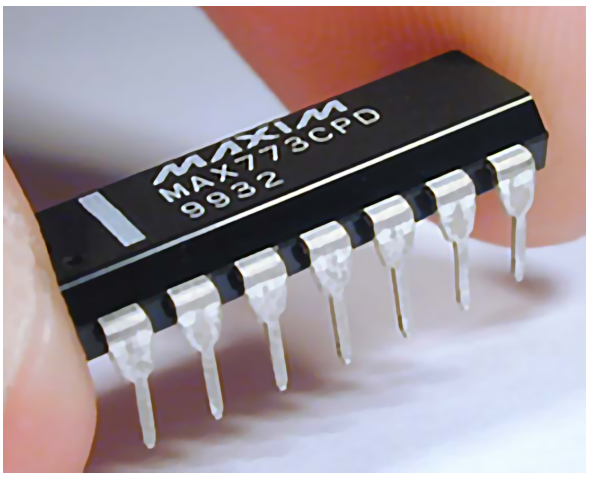
La terza immagine *"Chip"* è un'altra immagine spesso utilizzata nell'ambito della super-resolution. In questo caso non è stato possibile reperire un'originale sufficientemente grande da permettere un ripristino di risoluzione, quindi il test si limita all'aumento della risoluzione da *275x220* pixel a *1100x880* pixel. Diversamente dalle altre finora considerate, questa è un'immagine piuttosto regolare, non presenta molti dettagli fini (solo le creste delle impronte digitali), ma in compenso è la prima immagine considerata a contenere una scritta. Gli esperimenti sono quindi stati finalizzati ad ottenere immagini con una buona resa generale, senza però compromettere la qualità della scritta, in modo che sia il più leggibile possibile. In figura 5.13 è presentato il confronto fra le immagini ottenute e l'originale, mentre la tabella 3 riporta i tempi impiegati dai diversi algoritmi.



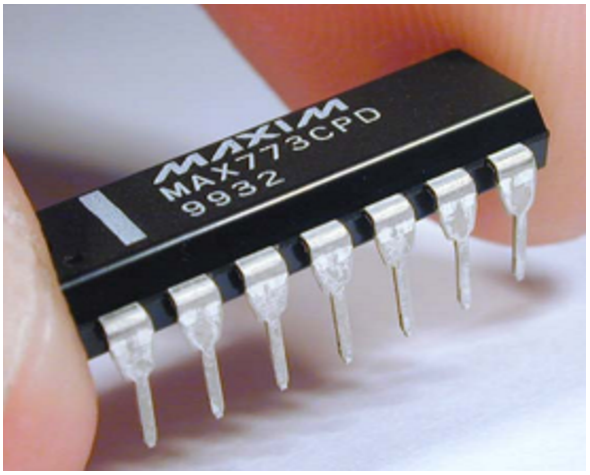
Originale



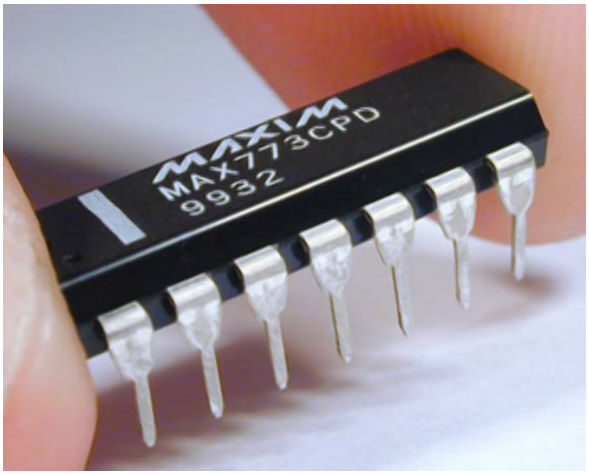
SR-CS DB4



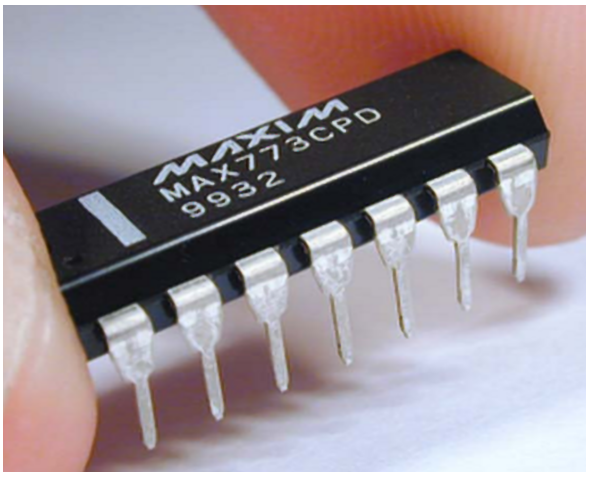
SR-CS Multigauss



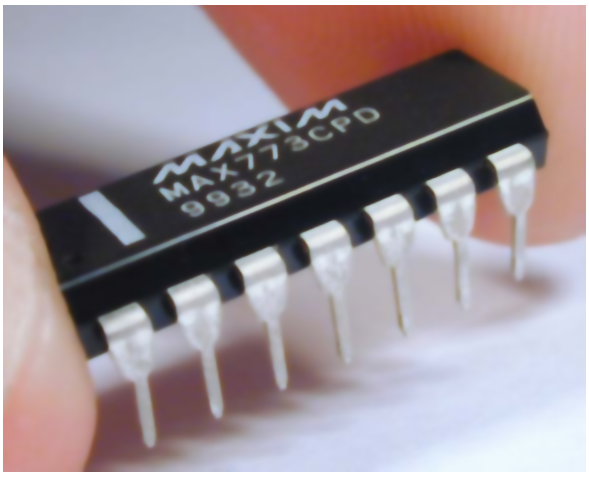
Interpolazione bicubica



NEDI



ICBI



SR-CS Gauss

Fig. 5.13: ingrandimento fattore 4. Parametri DB4 w = 0,05; Const = 0,00001; γ = 0,01; out = 4; in_max = 20. Parametri Multigauss: w = 1/50; filt_size = 5; σ = 1,5; out = 3.

Metodo	Tempo (sec)
SR-CS DB4	1515.687500
SR-CS Multigauss	1062.046875
Bicubica	0.078125
NEDI	593.296875
ICBI	1039.921875
SR-CS Gauss	688.734375

Tabella 3: tempi relativi all'aumento di risoluzione di "Chip" da 275x220 pixel a 1100x880 pixel.

Le due implementazioni presentate hanno risultati migliori degli altri algoritmi: i filtri DB4 restituiscono un risultato ben definito, la scritta è leggibile e non affetta da particolari artefatti o sfocature, inoltre non è visibile il rumore presente nei risultati degli altri metodi; i filtri gaussiani multi-resolution si comportano in maniera simile, ma esaltano ulteriormente il contrasto, senza quel senso di sfocatura presente anche nella versione DB4, inoltre i bordi risultano più lisci. È interessante notare che la versione con filtri gaussiani semplici usa gli stessi parametri che in [5] risultavano ottimi per questa immagine. Probabilmente il risultato sfocato è dovuto all'estrema suscettibilità del metodo SR-CS (indipendentemente dai filtri) rispetto il metodo di acquisizione. In figura 5.14, come per i test precedenti, sono riportati i dettagli in scala 1 a 1 per facilitare il confronto.

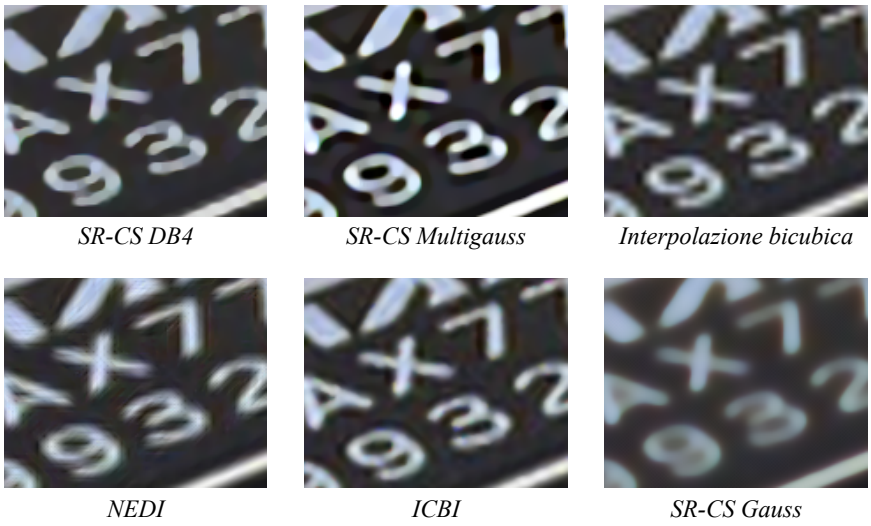
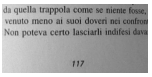
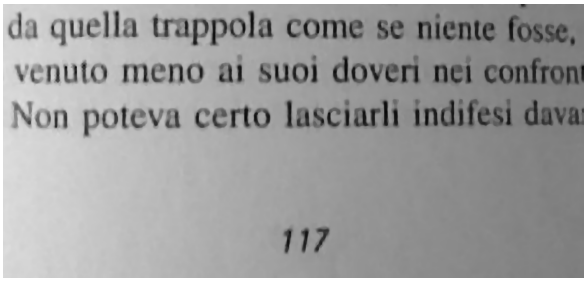


Fig. 5.14: confronto a dimensioni reali fra dettagli (150x120 pixel) degli ingrandimenti di fattore 4 presentati nella figura precedente.

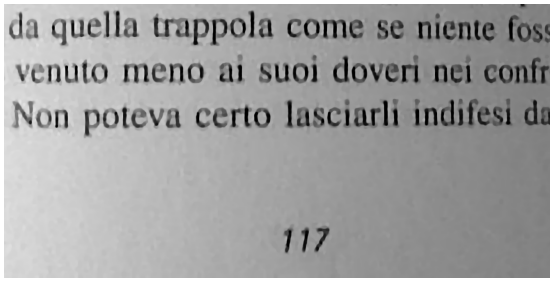
L'immagine successiva a differenza delle precedenti, è stata prodotta appositamente per questi test. Si tratta di un dettaglio di dimensioni *353x166* pixel estrapolato dalla fotografia della pagina di un libro e convertita in scala di grigi. Le scritte sono un soggetto interessante per questo genere di test, in quanto sono immagini sparse nel dominio del gradiente (le informazioni principali sono i bordi delle singole lettere), esattamente come il presupposto alla base dell'algoritmo presentato. Le principali difficoltà in questo caso sono date dallo sfondo non uniforme (un gradiente di colore) e dal notevole rumore introdotto dal sensore di piccole dimensioni. Anche in questo caso nei test si è cercato di privilegiare la leggibilità delle scritte. In figura 5.15 sono presentati i risultati relativi all'upscaling a *1412x664* pixel e in tabella 4 i tempi corrispondenti.



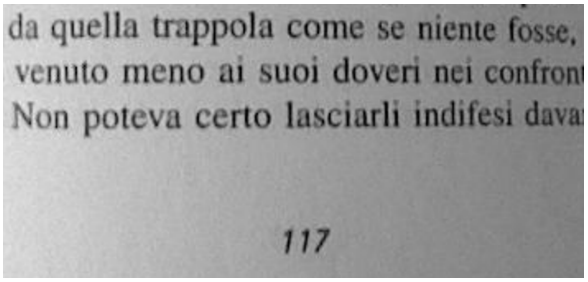
Originale



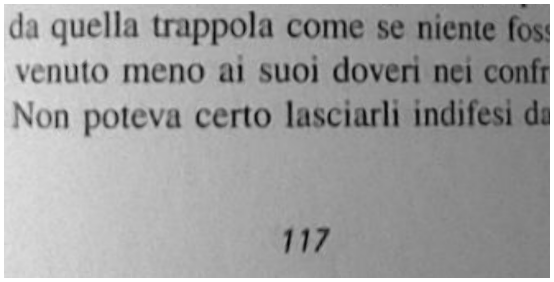
SR-CS DB4



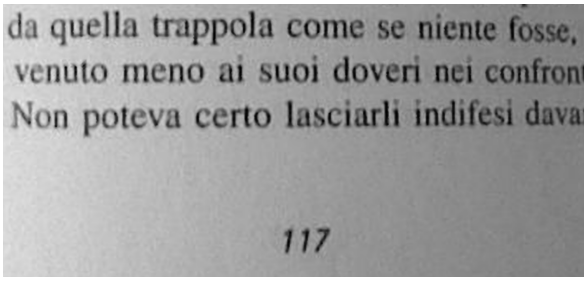
SR-CS Multigauss



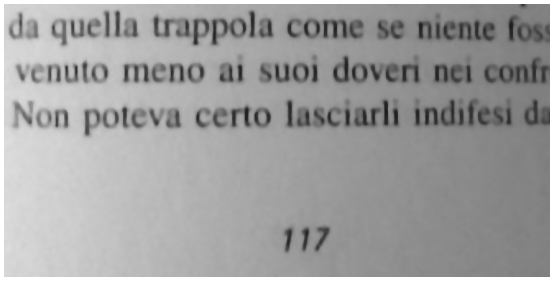
Interpolazione bicubica



NEDI



ICBI



SR-CS Gauss

Fig. 5.15: ingrandimento fattore 4. Parametri DB4 w = 0,05; Const = 0,00001; γ = 0,1; out = 3; in_max = 20 (in questo caso7 iterazioni di Chambolle). Parametri Multigauss: w = 1/50; filt_size = 5; σ = 1; out = 4.

Metodo	Tempo (sec)
SR-CS DB4	325.984375
SR-CS Multigauss	513.140625
Bicubica	0.125000
NEDI	196.937500
ICBI	301.234375
SR-CS Gauss	225.468750

Tabella 4: tempi relativi all'aumento di risoluzione di "Scritte" da 353x166 pixel a 1412x664 pixel.

In questo caso i due metodi proposti hanno i risultati migliori, in quanto riescono a restituire delle scritte leggibili, non sfocate e soprattutto non affette da rumore quanto i metodi di confronto. I filtri DB4 in particolare riducono notevolmente il rumore nella zona circostante alle lettere, inoltre la scritta risulta più definita. I filtri gauss multi-resolution presentano più rumore, ma comunque meno rispetto ai metodi di confronto, inoltre la scritta ha maggior contrasto ed è più adatta ad eventuali elaborazioni successive (ad esempio una binarizzazione per separare la scritta dallo sfondo). In figura 5.16 è possibile confrontare i dettagli dei risultati del test in scala 1 a 1.

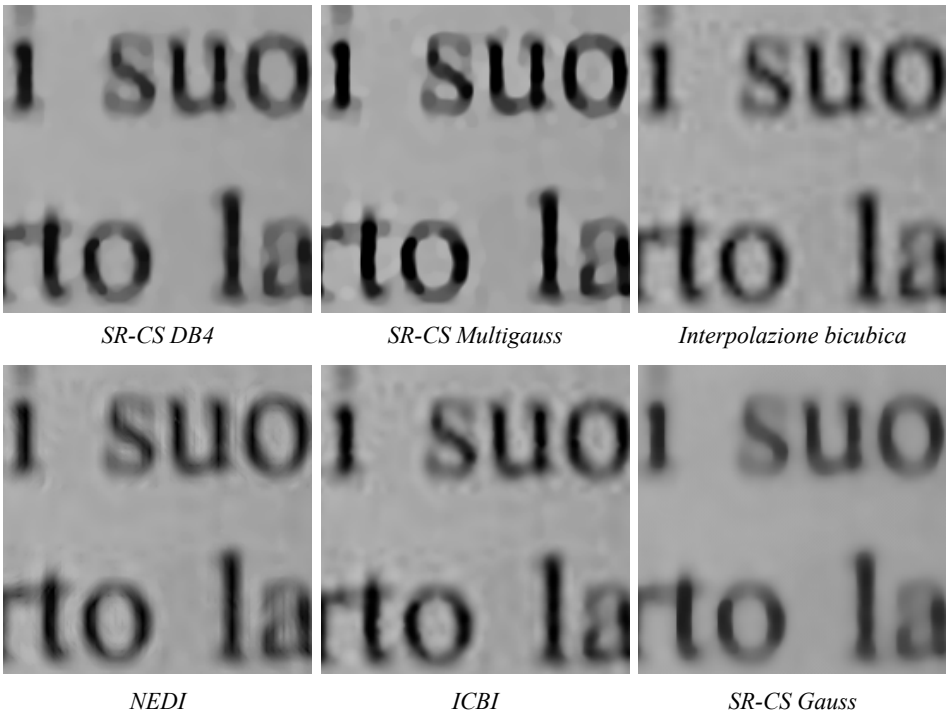


Fig. 5.16: confronto a dimensioni reali fra dettagli (180x180 pixel) degli ingrandimenti di fattore 4 presentati nella figura precedente.

L'ultima immagine considerata è la foto di una targa. Analogamente all'immagine precedente, è stata ottenuta estrapolando un dettaglio da una foto ottenuta con un sensore piccolo, ma in questo caso sono stati mantenuti tutti e tre i canali RGB. Anche in questo caso si è applicato un upscaling di fattore *4* da *107x33* pixel a *428x120* pixel, privilegiando la leggibilità della scritta e la regolarizzazione del rumore dovuto al sensore piccolo. In figura 5.17 sono presentati i risultati dei diversi metodi ed in tabella 5 i relativi tempi di esecuzione.

Fig. 5.17: ingrandimento fattore 4. Parametri DB4 w = 0,1; Const = 0,00005; γ = 0,01; out = 3; in_max = 20. Parametri Multigauss: w = 1/50; filt_size = 7; σ = 1,25; out = 3.

Anche in questo caso i due metodi presentati hanno risultati migliori, riducendo il rumore sullo sfondo. La versione migliore della scritta è ottenuta con i filtri gaussiani multi-resolution.

Metodo	Tempo (sec)
SR-CS DB4	67.656250
SR-CS Multigauss	29.296875
Bicubica	0.000000
NEDI	28.671875
ICBI	31.203125
SR-CS Gauss	18.078125

Tabella 5: tempi relativi all'aumento di risoluzione di "Targa" da 107x30 pixel a 428x120 pixel.

5.4 Ingrandimento di fattore 8

È stata eseguita un'ulteriore serie di test per verificare il funzionamento degli algoritmi per un fattore di ingrandimento 8. Secondo la teoria del campionamento compressivo esposta nel terzo capitolo, ed in particolare la disuguaglianza (3.10), per poter ricostruire correttamente un segnale, è necessario disporre di un numero minimo di misure. Purtroppo per un fattore 8, le informazioni di partenza saranno inferiori al numero minimo, rendendo dunque impossibile un ingrandimento diretto. Si è quindi deciso di spezzare l'ingrandimento in due fasi: un primo ingrandimento di fattore 2 ed un successivo ingrandimento diretto di fattore 4. Poiché non è necessario impostare le dimensioni per i filtri DB4, la relativa implementazione utilizza un unico set di parametri per entrambi i passi di upscaling; al contrario i filtri gauss multi-resolution necessitano di un'impostazione iniziale delle caratteristiche del filtro, quindi sono stati utilizzati parametri differenti per i due passi di upscaling.

La prima immagine considerata è un dettaglio di "Lena" di dimensioni $101x101$ pixel estrapolato da un originale di dimensioni $256x256$ pixel. Le figure 5.18 e 5.19 riportano i risultati dell'ingrandimento a $808x808$ pixel, la tabella 6 i relativi tempi.

Originale

SR-CS DB4

SR-CS Multigauss

Fig. 5.18: ingrandimento fattore 8. Parametri DB4 w = 0,1; Const = 0,00005; γ = 0,05; out = 4; in_max = 20. Parametri Multigauss: fattore d = 2; w = 1/10; filt_size = 5; σ = 1,5; out = 3; fattore d = 4; w = 1/10; filt_size = 7; σ = 2; out = 3.

Interpolazione bicubica

NEDI

ICBI

SR-CS Gauss

Fig. 5.19: risultati degli ingrandimenti di fattore 8 di "Dettaglio Lena" ottenuti con gli algoritmi di confronto.

Metodo	Tempo (sec)
SR-CS DB4	205.468750
SR-CS Multigauss	227.875000
Bicubica	0.031250
NEDI	141.312500
ICBI	123.515625
SR-CS Gauss	206.156250

Tabella 6: tempi relativi all'aumento di risoluzione di "Dettaglio Lena" da 101x101 pixel a 808x808 pixel.

I filtri DB4 riescono a mantenere un certo livello di dettaglio, comparabile ai risultati degli altri metodi, seppur con qualche scalettatura sui bordi, in compenso l'immagine è più regolare e priva di rumore, soprattutto nella pelle del viso. I filtri gaussiani multi-resolution hanno bordi più lisci e contrasto maggiore, perdono qualche dettaglio ma il risultato è comunque buono.

La seconda immagine è un dettaglio dell'immagine "Bambino", sempre di dimensioni *101x101* pixel ed estrapolata da una versione originale di dimensioni *256x256* pixel. Nelle figure 5.20 e 5.21 sono presentati i risultati del test ed in tabella 7 i relativi tempi.

Originale

SR-CS DB4

SR-CS Multigauss

Fig. 5.20: ingrandimento fattore 8. Parametri DB4 w = 0,05; Const = 0,00001; γ = 0,05; out = 4; in_max = 20. Parametri Multigauss: fattore d = 2; w = 1/50; filt_size = 5; σ = 1,5; out = 3; fattore d = 4; w = 1/10; filt_size = 7; σ = 2; out = 3.

Anche in questo caso il filtro DB4 permette di ottenere dettagli in maniera analoga ai metodi di confronto, accettando una certa quantità di scalettatura, mentre i filtri gaussiani permettono di ottenere bordi molti più lisci e definiti. Entrambi i metodi proposti hanno risultati migliori per quanto riguarda il disturbo presente sull'immagine originale: la pelle è più regolarizzata ed appare più liscia, senza il rumore presente negli altri metodi, senza per questo sembrare innaturale.

Interpolazione bicubica

NEDI

ICBI

SR-CS Gauss

Fig. 5.21: risultati degli ingrandimenti di fattore 8 di "Dettaglio Bambino" ottenuti con gli algoritmi di confronto.

Metodo	Tempo (sec)
SR-CS DB4	819.218750
SR-CS Multigauss	683.578125
Bicubica	0.187500
NEDI	407.859375
ICBI	330.781250
SR-CS Gauss	461.203125

Tabella 7: tempi relativi all'aumento di risoluzione di "Dettaglio Bambino" da 101x101 pixel a 808x808 pixel.

La terza immagine è un dettaglio della scritta dell'immagine "Chip" estrapolato da un originale di dimensioni *550x440* pixel. Anche in questo caso la dimensione dell'immagine è *101x101* pixel e viene portata alle dimensioni finali di *808x808* pixel. Le figure 5.22 e 5.23 riportano le immagini risultanti dal test, mentre in tabella 8 sono trascritti i tempi di esecuzione.

Originale

SR-CS DB4

SR-CS Multigauss

Fig. 5.22: ingrandimento fattore 8. Parametri DB4 w = 0,1; Const = 0,00005; γ = 0,05; out = 4; in_max = 20. Parametri Multigauss: fattore d = 2; w = 1/50; filt_size = 5; σ = 1; out = 2; fattore d = 4; w = 1/10; filt_size = 7; σ = 2; out = 3.

Come nel caso dell'esperimento sull'ingrandimento di fattore *4*, le immagini prodotte dai metodi proposti risultano più regolari e non affette da rumore rispetto a quelle prodotte dagli altri metodi.

Metodo	Tempo (sec)
SR-CS DB4	583.984375
SR-CS Multigauss	677.656250
Bicubica	0.062500
NEDI	416.812500
ICBI	310.406250
SR-CS Gauss	459.390625

Tabella 8: tempi relativi all'aumento di risoluzione di "Dettaglio Chip" da 101x101 pixel a 808x808 pixel.

Interpolazione bicubica

NEDI

ICBI

SR-CS Gauss

Fig. 5.23: risultati degli ingrandimenti di fattore 8 di "Dettaglio Chip" ottenuti con gli algoritmi di confronto.

La prossima immagine ad essere considerata è invece un'immagine sintetica prodotta da un computer e raffigurante un ideogramma. Come le immagini delle scritte già analizzate, anche questa rispetta il presupposto di essere sparsa nel dominio del gradiente, ma inoltre ha la caratteristica di essere un'immagine binaria: la scritta è completamente nera, senza sfumature e lo stesso vale per lo sfondo, completamente bianco. Inoltre proprio per il fatto di essere prodotta artificialmente è sicuramente priva di qualunque difetto legato a rumore o sfocature. Queste caratteristiche la rendono particolarmente adatta a testare le caratteristiche degli algoritmi presentati in questo lavoro. L'immagine di input e di

dimensioni *72x72* pixel e viene portata a *576x576* pixel. I risultati del test sono riportati nelle figure 5.24 e 5.25, mentre i tempi di esecuzione nella tabella 9.

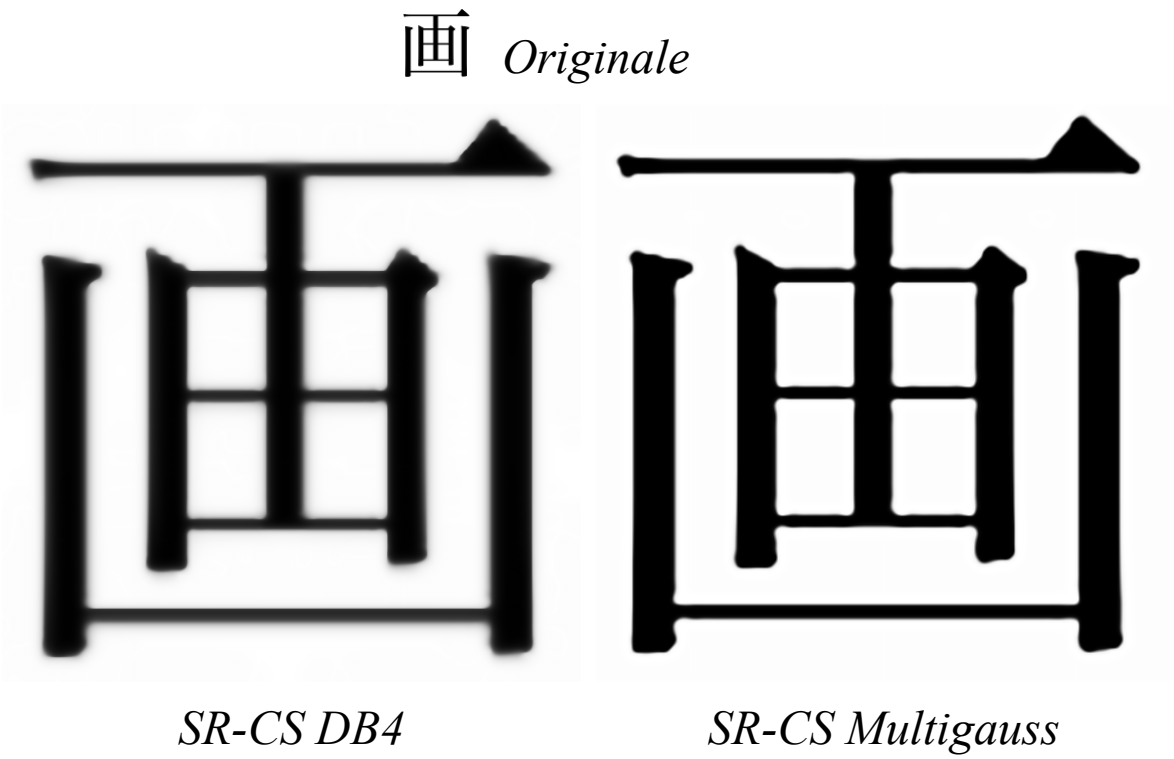


Fig. 5.24: ingrandimento fattore 8. Parametri DB4 w = 0,05; Const = 0,00005; γ = 0,01; out = 4; in_max = 20. Parametri Multigauss: fattore d = 2; w = 1/30; filt_size = 7; σ = 1,25; out = 3; fattore d = 4; w = 1/30; filt_size = 7; σ = 1,25; out = 3.

Questo test fa emergere i principali difetti dei metodi di confronto: il problema è mal condizionato per NEDI che non riesce a completare l'interpolazione; ICBI, che è stato concepito per immagini naturali, presenta delle ombre in prossimità dei bordi; l'interpolazione bicubica mostra la sua naturale sfocatura; SR-CS con filtri gaussiani ha bordi netti, ma risultano "rovinati" dal filtro. Anche il metodo basato su filtri DB4 presenta un alone notevole attorno ai bordi. Attraverso gli esperimenti si è appurato che modificare i parametri in modo da aumentare il contrasto, non migliora la qualità dell'immagine, anzi la peggiora, rendendo ancora più definito l'alone che si presenta come una macchia grigia attorno ai tratti. Questo difetto è probabilmente dovuto alla natura stessa dei filtri DB, che essendo composti sempre da un numero pari di elementi, non possono essere centrati sulla diagonale della matrice di Toeplitz. Questo introduce un'inevitabile traslazione nell'immagine finale e questi aloni. Proprio per questo difetto è stata sviluppata la versione con filtri gaussiani multi-resolution, che essendo centrati e simmetrici permettono di ottenere risultati migliori in questo genere di test.

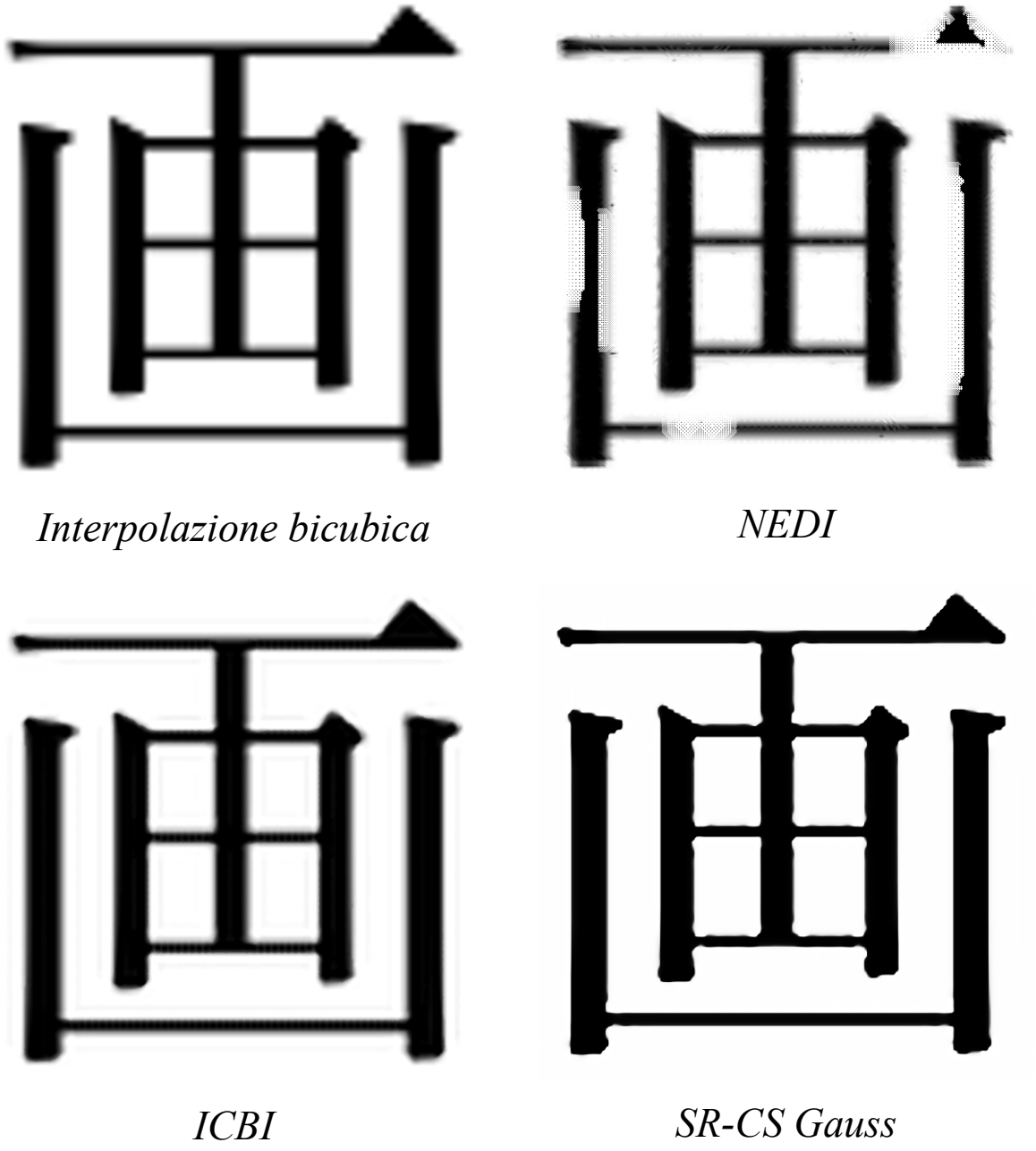


Fig. 5.25: risultati degli ingrandimenti di fattore 8 di "Ideogramma" ottenuti con gli algoritmi di confronto.

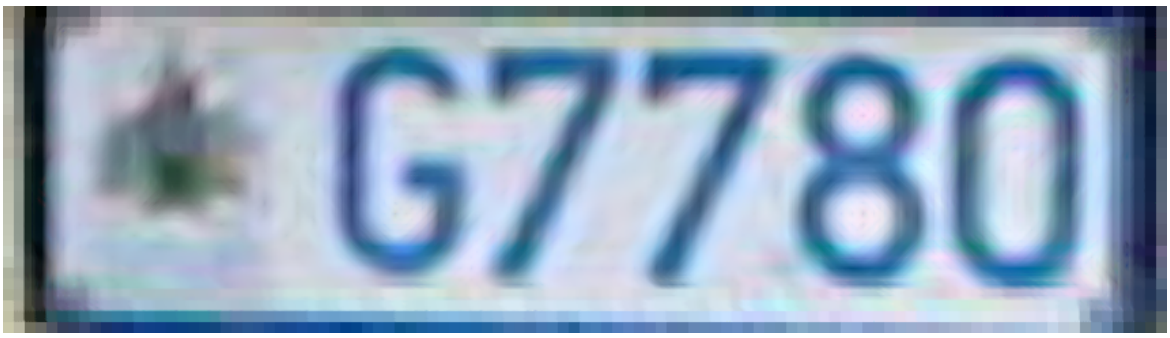
Metodo	Tempo (sec)
SR-CS DB4	106.015625
SR-CS Multigauss	102.421875
Bicubica	0.000000
NEDI	164.859375
ICBI	143.359375
SR-CS Gauss	92.046875

Tabella 9: tempi relativi all'aumento di risoluzione di "Ideogramma" da 72x72 pixel a 576x576 pixel.

L'ultimo test è stato eseguito sull'immagine della targa, portandola da dimensioni *107x30* pixel a *856x240* pixel. I risultati sono mostrati nelle figure 5.26 e 5.27 ed i relativi tempi di elaborazione in tabella 10.

Fig. 5.26: ingrandimento fattore 8. Parametri DB4 w = 0,1; Const = 0,00005; γ = 0,01; out = 3; in_max = 20. Parametri Multigauss: fattore d = 2; w = 1/10; filt_size = 7; σ = 1; out = 2; fattore d = 4; w = 1/80; filt_size = 7; σ = 2; out = 3.

Anche in questo caso i due metodi proposti ottengono i risultati migliori, regolarizzando il rumore presente nella fotografia originale, ma è soprattutto il metodo con filtri gaussiani multi-resolution ad isolare al meglio la scritta dallo sfondo (sebbene i bordi risultino leggermente deformati).



NEDI

ICBI

SR-CS Gauss

Fig. 5.27: risultati degli ingrandimenti di fattore 8 di "Targa" ottenuti con gli algoritmi di confronto.

Metodo	Tempo (sec)
SR-CS DB4	247.245985
SR-CS Multigauss	182.328125
Bicubica	0.093601
NEDI	93.319798
ICBI	141.336906
SR-CS Gauss	130.775638

Tabella 10: tempi relativi all'aumento di risoluzione di "Targa" da 107x30 pixel a 856x240 pixel.

Per quanto riguarda i tempi di esecuzione, i metodi proposti impiegano in genere molto più tempo degli altri algoritmi di confronto considerati, anch'essi implementati con Matlab. Bisogna considerare che l'algoritmo SR-CS è stato implementato tenendo in considerazione la leggibilità del codice e la facilità di modifica, a scapito della velocità di esecuzione. Certi accorgimenti adottati, come le numerose chiamate a funzioni esterne, nel contesto di un linguaggio interpretato rallentano notevolmente il codice. Sicuramente una futura implementazione con un linguaggio compilato potrebbe garantire risultati migliori dal punto di vista dei tempi di elaborazione.

Conclusioni

In questa tesi è stato proseguito il lavoro cominciato in [5]: la teoria del compressed sensing applicata all'ambito della super-resolution è stata integrata con il concetto di multi-resolution. I metodi implementati ottengono risultati comparabili ad altre tecniche utilizzate per l'upscaling ed a volte le immagini prodotte presentano addirittura caratteristiche migliori. Attraverso gli esperimenti si è inoltre evidenziato che l'approccio con filtri Daubechies, nonostante le buone prestazioni, presenti dei limiti dovuti alla loro stessa natura. In compenso l'approccio a molti livelli di risoluzione applicato a differenti tipologie di filtro si è dimostrato essere un'idea valida, che semplifica l'impostazione iniziale dei parametri a seconda del livello di risoluzione desiderato. Nel caso specifico sono stati utilizzati dei filtri gaussiani, ma la tecnica è applicabile anche ad altre tipologie di filtri di blur. Questo approccio merita di essere ulteriormente approfondito in quanto potrebbe rivelarsi molto utile nella stima del blur applicato all'immagine, infatti uno dei principali difetti dell'algoritmo è la difficoltà nell'impostazione dei parametri per ottenere la miglior ricostruzione possibile. Questo è dovuto al fatto che la fase di acquisizione resta comunque un'incognita. Un possibile sviluppo futuro di questo approccio è dunque l'integrazione dell'algoritmo SR-CS con tecniche di blind-deconvolution, che permettano di stimare in maniera automatica l'operatore di convoluzione applicato in fase di acquisizione, riducendo così la necessità di intervento da parte dell'utente.

Bibliografia

[1] G.Forti, *"Fotografia Teoria e pratica della reflex"*, Editrice Reflex, Roma, pp. 114-121, 2010.

[2] AAVV, *"Image resolution"*, http://en.wikipedia.org/wiki/Image_resolution, 2012.

[3] AAVV, *"Camera lens"*, http://en.wikipedia.org/wiki/Camera_lens, 2012.

[4] AAVV, *"Superresolution"*, http://en.wikipedia.org/wiki/Superresolution, 2012.

[5] T.Giorgetti, *"Un approccio basato sulla teoria del Compressed Sensing per la Super Resolution di immagini"*, Tesi di Laurea in Scienze dell'Informazione, Università di Bologna, A.A. 2010/2011.

[6] S.C.Park, M.K.Park, M.G.Kang, *"Super-Resolution Image Reconstruction: A Technical Overview"*, IEEE Signa Processing Magazine, pp. 21-36, Maggio 2003.

[7] X.Li, M.T.Orchard, *"New Edge Directed Interpolation"*, IEEE Transactions on Image Processing, 2001.

[8] N.Asuni, *"iNEDI Tecnica Adattiva per l'Interpolazione di Immagini"*, Tesi di Laurea in Tecnologie Informatiche, Università degli Studi di Cagliari, A.A. 2006/2007.

[9] A.Giachetti, N.Asuni, *"Fast artifact free image interpolation"*, Proc. Brit. Machine Vis. Conf. (BMVC), 2008.

[10] D.Glasner, S.Bagon, M.Irani, *"Super-Resolution from a Single Image"*, IEEE 12th Int Conf. on Computer Vision, pp. 349 - 356, Ottobre 2009.

[11] G.Santi, *"Metodi numerici per ridurre i tempi di esposizione durante l'acquisizione di immagini mediche"*, Tesi di Laurea in Scienze dell'Informazione, Università di Bologna, A.A. 2008/2009.

[12] G.Pope, *"Compressive Sensing A Summary of Reconstruction Algorithms"*, Tesi di laurea in Informatica, Swiss Federal Institute of Technology, Zurigo, A.A. 2007/2008.

[13] J.Romberg, M.Wakin, *"Compressed Sensing: A Tutorial"*, http://people.ee.duke.edu/~willett/SSP//Tutorials/ssp07-cs-tutorial.pdf, 2007.

[14] R.Vershynin, *"Compressed sensing"*, http://www-personal.umich.edu/~romanv/slides/2007-UCD-CS.pdf, 2007.

[15] D.Mackenzie, *"Compressed sensing makes every pixel count"* in: *"What's happening in mathematical sciences"*, American Mathematical Society, Providence, vol. 7, pp. 114-127, 2009.

[16] T.Tao, *"Compressed sensing and single-pixel cameras"*, http://terrytao.wordpress.com/2007/04/13/compressed-sensing-and-single-pixel-cameras/, 2007.

[17] T.Tao, *"Compressed sensing - Or: the equation Ax=b revisited"*, http://terrytao.files.wordpress.com/2009/08/compressed-sensing1.pdf, 2009.

[18] P.L.Combettes, V.R.Wajs, "Signal Recovery by Proximal Forward-backward Splitting" SIAM Journal on Multiscale Modelling and Simulation, vol. 4, pp. 1168-1200, 2010.

[19] A.Beck, M.Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems", SIAM J. Imaging Science, vol. 2, no. 1, pp. 183-202, 2009.

[20] L.B.Montefusco, D.Lazzaro, *"An iterative L_1-based image restoration algorithm with adaptive parameter estimation"*, IEEE Transaction On Signal Processing, Ottobre 2011.

[21] H.Rauhut, *"Circulant and Toeplitz matrices in compressed sensing"*, SPARS'09 - Signal Processing with Adaptive Sparse Structured Representations, 2009.

[22] T.S.Cho, S.Paris, B.K.P.Horn, W.T.Freeman, *"Blur Kernel Estimation using the Radon Transform"*, In proceeding of: The 24th IEEE Conference on Computer Vision and Pattern Recognition, Colorado Springs, Giugno 2011.

[23] S.Mehta, B.Li, *"Fast Global Blur Estimation Using a Learning-based Approach"*, International Workshop on Video Processing and Quality Metrics, Gennaio 2009.

[24] G.Casciola, L.B.Montefusco, S.Morigi, *"Edge-driven Image Interpolation using Adaptive Anisotropic Radial Basis Functions"*, Journal of Mathematical Imaging and Vision, Volume 36, Issue 2, pp. 125 - 139, Febbraio 2010.